# MANUALE SPRIX

# SPRIX COLLAGE

# MANUALE SPRIX/COLLAGE

Manuale operativo per gli ambienti di sviluppo sprix e Collage.

**Per ricercare le voci presenti nel manuale sprix, all'interno del sorgente i comandi devono essere scritti senza il simbolo "$" finale e senza il simbolo funzione"()"**

I punti di seguito elencati rappresentano i capitoli principali dell'editor e dell'ambiente operativo

# 1 – CARATTERISTICHE

Sprix e Collage sono strumenti di sviluppo semplici e potenti, integrati nel Gestionale Passepartout per personalizzarlo in base alle specifiche esigenze dell'Utente: dall'accesso a dati in scrittura, lettura e stampa con apposite istruzioni, funzioni e variabili a sofisticate elaborazioni/procedure Sprix, anche attraverso accessi a dati provenienti da fonti esterne; dagli interventi sul Gestionale per leggerne, modificarne o integrarne i contenuti visivi e di stampa o simulare o condizionare l'operatività dell'Utente tramite la programmazione ad eventi Collage, fino alla combinazione dei programmi Sprix e Collage quando utile (es. l'esecuzione di uno Sprix da un evento Collage).

Il Gestionale non è necessariamente da sviluppare. Le ampie e affidabili soluzioni standard offerte tutelano sempre l'utente e non sono modificabili da eventuali configurazioni della customizzazione. La personalizzazione è verificata dalla piattaforma con un compilatore integrato non vincolato al sistema operativo e viene resa trasparente al sistema e senza obbligo di manutenzioni per l'aggiornamento del Gestionale, di cui non raggiunge la potenza.

L'utente non è vincolato e può rimuovere o sostituire le personalizzazioni senza ripercussioni sull'integrità dei dati e le funzionalità di base, nemmeno cambiando il sistema e configurazione o livello di prodotto, che rendessero utile un adeguamento alla nuova situazione.

Sprix è un linguaggio di programmazione simile al Basic, suddiviso in routine di codice elaborato riga per riga. Per sviluppare in ambienti e linguaggi più evoluti è disponibile la Suite dei componenti Shaker, che all'occorrenza non tolgono la possibilità di effettuare verifiche in Sprix.

# 2 – REQUISITI

La programmazione in Sprix e Collage richiede l'installazione del Gestionale con relativa licenza di sviluppo. Richiede inoltre la visione della documentazione del disco di rilascio: nella directory principale si trovano il leggimi di versione ed il manuale Utente; nella sottocartella '\gest\sprix' si trovano degli eseguibili per le procedure particolari e nella sottocartella 'gest\doc\conces' si trovano il manuale di Sprix e i tracciati, le tabelle, diagrammi,... Ulteriori informazioni di approfondimento sono a disposizione su EasyPass e ricercabili tramite parole chiave.

Tanto maggiori sono le capacità di programmazione, di analisi del problema e di preparazione sul Gestionale, quanto migliori sono i risultati che si ottengono, senza replicare funzioni standard già disponibili e senza incontrare eventuali vincoli di varia natura, anche commerciali; sono disponibili di diversi livelli di licenza di sviluppo [F1+F1+F5] ed indipendentemente da essi le funzioni Sprix disponibili sono direttamente relazionate alla configurazione del prodotto acquistato. Esempio, un prodotto contabile potrebbe non eseguire funzioni di magazzino o viceversa.

Gli argomenti contenuti nel manuale Utente integrano il manuale Sprix, sia a livello sistemistico e di configurazione (dalle caratteristiche dei componenti Client-Server e loro remotizzazioni e reindirizzamenti, alla gestione degli accessi utente e restrizioni di accesso a directory file/seriali, dalle esecuzioni batch e shell remote alle operazioni su porte di comunicazione di entità Client-Server non autorizzate fino alla definizione delle stampanti) sia a livello di natura e logica di gestione del dato e di operatività sul Gestionale (contabilità, magazzino, trasmissione archivi,...), sia di ordinaria amministrazione che straordinaria (come le conversioni a nuove o diverse funzioni). Le registrazioni richiedono dati che possono variare a seconda delle circostanze e per conoscerli, basta registrare nel Gestionale, leggere con Sprix e verificare eventualmente col debugger.

Sprix e Collage nascono dal Gestionale e sono ad esso subordinati per funzioni, controlli, permessi ed accessi archivi, semaforica, quantità di dati in ' cache ' (da azzerare per le nuove elaborazioni). Il Gestionale riorganizza automaticamente, o tramite Servizi, i dati in caso di cancellazioni, mentre Sprix deve replicare queste logiche di ricostruzione massiva, pena warning / errori non bloccanti. Esempio, cancellazione di record o elementi di strutture articolate, senza azzerare, inizializzare o riordinare TUTTE le variabili coinvolte nell'operazione, anche se NON appartenenti ai record cancellati. I controlli effettuati su Sprix riguardano solo le operazioni non effettuabili nemmeno dal Gestionale. Esempio, cancellazione di un cliente movimentato e/o di una fattura già pagata.

# 3 – GESTIONE PROGRAMMI

Dal menu 'Servizi-Personalizzazioni-Sprix/Collage’ si accede all'ambiente di sviluppo per singola omonima voce di menu del gestionale; per ciascuna voce è possibile realizzare fino a 99 programmi Sprix che vengono attivati in un apposito sottomenu del menu opportuno. La voce Collage è dedicata per i programmi Collage che si attivano con un codice di menu nell'elenco all'interno della voce. La voce Libreria è un archivio di librerie in ausilio a tutti i programmi Sprix e Collage. L'ultima voce è una funzione di stampa delle variabili disponibili nelle tabelle di lettura.

Selezionando una voce si apre un elenco scorribile, in cui selezionare e gestire ciascun programma, accedendovi attraverso l'ingresso nell'EDITOR. Il numero a sinistra indica il numero di programma relativo alla voce selezionata e l'elenco rappresenta l'esposizione dei programmi nel sottomenu della voce e che è personalizzabile [F5]; negli ambienti in cui è consentito ciò non cambia il tasto di selezione rapida 'X' del sottomenu Sprix (es. in Contabilità 'Alt+C+X'). Se consentito dall'ambiente, anche l'inserimento di un solo carattere maiuscolo nel 'nome del formato’, cioè quello di ingresso e di battesimo del programma (nome libero e non univoco), assume funzione di selezione rapida. È sconsigliato l'uso di prefissi con caratteri speciali. La lista dei programmi è memorizzata nel file 'sprix000' presente nella directory dei file dei programmi Sprix e Collage indicata nel prospetto di directory visualizzabili con la sequenza dei tasti [F1+F1+F8].

Se è previsto dalla tipologia di architettura dei dati aziendali del Gestionale, la directory è specificata nel profilo anagrafico aziendale del menu "Azienda" ed è accessibile con tasto [F8] per essere dichiarata liberamente in forma estesa con carattere '\' terminale oppure nella forma breve '+sig\' che assume la directory dati dell'azienda 'sig' (cioè il default in assenza di dichiarazione) che viene fisicamente creata ed eventualmente modificabile con [F5] anche oltre i 32 caratteri consentiti (sconsigliato); sono memorizzabili più directory [F4], selezionabili con [F2], cancellabili con [ShF3].

Il Gestionale che non consente la gestione della directory nel profilo anagrafico aziendale, significa che ha vincoli su directory (dati e programmi). Questo tipo di gestione, stabilisce l'accesso dell'azienda a determinati programmi Sprix attraverso l'appartenenza o meno a un determinato Gruppo Sprix (v. menu Anagrafica Azienda). L'azienda può appartenere a un gruppo che va da zero (la directory generale degli sprix "\pers\sprix") a 99 (una specifica sottodirectory "\pers\sprix\sprix.nnn", dove nnn può assumere un valore da 1 a 99). Dalla voce Utenti/Chiavi d'accesso del menu Servizi è inoltre possibile stabilire, che il gruppo utenti acceda direttamente a determinati Sprix in "modalità solo Sprix", cioè senza l'accesso ad altri menu dell'azienda assegnata agli utenti.

Proseguendo con Invio si accede all'editor di testo, ma prima di descriverlo ne usciamo con Esc, ottenendo una voce in elenco composta da tre elementi secondo la seguente sintassi: 'nnn','nome del formato del programmà,'spxnnn'. Il terzo elemento identifica il nome univoco del file del programma, la cui  sintassi di default è in base all'elencazione della voce selezionata, ma è anche personalizzabile con la compilazione. Questo nome del file è generato nella directory dei programmi Sprix e Collage con estensione sorgente '.ss' in quanto in esso è contenuto il codice che tramite compilazione genera un omonimo file con estensione eseguibile '.es' per l'esecuzione del programma anche da parte di più utenti indipendentemente dal sorgente, ma che è aggiornabile da un utente alla volta solo attraverso di esso, non esistendo al cuna possibilità di decompilazione.

## 4 – COMPILAZIONE

Il nome file ha max 8 caratteri esclusa l'estensione e, a parte i comandi di sistema fra cui il più importante è quello della copia di sicurezza della intera directory sprix, l'amministrazione dei file avviene tramite l'elenco e tramite la funzione di COMPILAZIONE [F10]: dopo avere generato un file fittizio in una posizione fittizia dell'elenco, premendo il pulsante 'Nome File' [F4] è possibile acquisire il file sorgente direttamente in elenco e se si tratta del solo eseguibile, per concludere l'installazione è necessario tentare di accedere nell'editor e confermare il messaggio di 'creazione sorgente non possibile; dal elenco è possibile cancellare fisicamente il file con[ShF3]; la modifica nel compilatore [F10] dei campi "Menu" e "Descrizione"  duplica il nome del formato in un'altra posizione, che resta collegata allo stesso sorgente sempre chè non avvenga la contestuale rinomina, anche personalizzata, del campo 'Nome File' per duplicarne anche il file sottostante; se non avviene anche questa seconda operazione l'eventuale cancellazione della posizione originaria va preceduta da una modifica fittizia nell'elenco del 'Nome File' [F4] per evitare la perdita del sorgente. Il compilatore controlla e chiede conferma sul rischio di sovrascrittura del file su un altro con lo stesso nome nella stessa directory. Il pulsante 'Nome File' [F4] è anche utile per lo spostamento del nome del formato nell'elenco. In caso sia attivo MDS nell'installazione tramite la pressione del tasto ShF3 viene richiesto se si vuole cancellare solo il sorgente. In caso di risposta negativa verranno cancellati sia il sorgente che l'eseguibile. In caso di risposta affermativa sarà cancellato solo il file sorgente ".ss".

## 5 – EDITOR

Dalle posizioni negli elenchi dei programmi con Invio si accede nell'EDITOR per gestire il codice sorgente Sprix. Dei due assi cartesiani sono indicate le posizioni di riga e colonna del cursore. La riga verticale dell'asse Y è il separatore che caratterizza la tipologia di riga, che quando è contrassegnata con un punto (.) significa che è VUOTA; premendo una sequenza di tasti [Invio] si contrassegna la riga (e quelle successive) con un simbolo 'N' che stabilisce che la riga è di tipo STAMPA (di quanto definito su di essa e di cui si approfondirà nella sezione relativa alle stampe). Per rendere la riga di tipo OPERATIVA (cioè per effettuare le operazioni) è necessario cancellare il simbolo premendo [F5+Canc+Invio] rendendo la riga non marcata da alcunché (per ripristinare ripetere l'operazione [F5+tasto N+Invio]).

Nella riga operativa possono inserirsi comandi con espressioni, istruzioni, funzioni interpretate dal programma anche in sequenza separate dal carattere ':', ma rese non operative se precedute dal carattere apice (') che trasformano in commento tutto ciò che viene dopo di esso fino alla riga successiva (anche l'istruzione REM commenta, ma a differenza dell'apice si posiziona solo ad inizio riga). La lunghezza della riga è 220 caratteri (ASCII 32-126) più una stringa di fine riga. Le righe sono 3000 e sono superabili con il riutilizzo del codice attraverso:

## 6 – TASTI EDITAZIONE

La pressione nell'editor del tasto [Alt+F9] elenca i tasti funzione attivi, del tasto [F1] l'help in linea quando previsto, dei tasti [ShF1] del manuale  di sprix.

Il tasto [F12] consente di IMPORTARE il codice di un sorgente da locale (con 'L') o da remoto (con 'S'); l'import è quindi consentito sia dalla stessa directory in cui risiedono i file sorgenti sprix (directory di configurazione del gestionale oppure, obbligatoriamente la directory '\pers' se prevista obbligatoriamente dalla componente Server; in entrambi i casi si indica solo il nome del file Sprix/Collage da importare), sia da una directory diversa e remota (es. da quella del Client in locale verso la directory '\pers' del Server remoto). Quando consentito, directory e file sono navigabili con appositi tasti funzione (in genere F2,F3,...).

Nota Bene: l'operazione non importa la funzione PARAMETRI DI ESECUZIONE [F6] e nemmeno file di libreria che sono eventualmente richiamati tramite #APPENDISPX (vanno importati a parte).

Oltre al import di sorgente si può importare del testo da file (con 'T'): occorre indicare la directory estesa del file testo (importato con caratteristiche della riga marcata e compatibilmente a quelle dell'editor) e nel secondo campo indicare codici ASCII dei Caratteri Fine Record (CFR) del file, con default pari a 13+10 cioè il Carriage Return+Line Feed (CR+LN) del sistema Windows (mentre in ambiente Linux è 10), che se incontrati fanno proseguire l'import su nuova riga (come avviene in caso di testo eccedente la lunghezza dell'editor); caratteri non consentiti sono sostituiti da blank (32); file Sprix esportati in formato 'ASCII' e importati 'T' interpretano la stampa del dizionario come una costante (come indicata nell'editor). I tasti funzione [F7-F8] scorrono il codice a sinistra/destra di 76/94 caratteri e [ShF8] va a fine riga del codice. [F11] ricerca la stringa di testo (o numero di riga con sintassi ':..numerò), con o senza attivazione [F2] di differenze fra MAIUSCOLO e minuscolo; richiamando la funzione si può ricercare in su/giù.

Infine [ShF11] trasforma i caratteri da minuscolo a minuscolo o viceversa, anche in sequenza finché viene tenuto premuto il tasto.

# 7 – COMPILATORE [F10] E DEBUGGER [SHF12]

La conferma con [F10] apre il COMPILATORE, con l'indicazione del menu, posizione, nome del formato e del file senza estensione, tutti modificabili e in fine la possibilità di stampare il sorgente ed eventualmente anche la lista finale delle variabili utilizzate con relativa locazione. Nella maschera che si apre con F10 è possibile stabilire il tipo di operazione da eseguire sul sorgente. L'operazione 'È' consente di esportare il sorgente senza compilazione: questa scelta attiva un campo in cui indicare il Percorso di destinazione (la directory default in locale e "<c:\tmp\"); il nome del file assume come default la sintassi numerica stabilita dai valori dei campi precedenti (Salva nel menu e Numero Sprix), ma è modificabile.

A differenza della compilazione di tipo 'S'-'D', quella 'N' non controlla la correttezza della sintassi per generare l'eseguibile e pertanto, serve per effettuare salvataggi del sorgente, dato che essendo aperto è danneggiabile da eventuali inconvenienti al sistema. La compilazione 'S' senza errori gene ra l'eseguibile pronto per essere eseguito dal menu o anche direttamente dal sorgente aperto tramite [ShF12] e cioè, l'esecuzione in editazione con controllo di compilazione (non consentita in Collage). La compilazione 'D' è il DEBUGGER per ricercare gli errori attraverso l'esecuzione del programma 'passo a passo' da menu o con [ShF12], per verificare ad ogni 'passo' eseguito con [F10] il flusso logico del programma, il valore delle variabili d'interesse e stabilire break point d'interruzione del programma per verificare la riga prossima all'esecuzione. La finestra di debugger copre l'esecuzione del programma, ma è temporaneamente chiudibile con il tasto 'schermò[F4] per verificarne il risultato a video, analogamente a quanto accade quando il programma prevede un'immissione da tastiera, posizionando il cursore dal debugger alla finestra d'immissione (attiva). La finestra di debugger ha due sezioni, una superiore con la porzione di codice scorribile con frecce [su / giu] e con la visualizzazione in reverse di quella prossima all'esecuzione e una inferiore per visualizzare i 'comandi' debugger inseribili nel campo di immissione (richiamabili inserendo 'h' o 'help') e sono i seguenti:

- '?' visualizza il valore attuale di una variabile (se array l'elemento de ve essere una costante, es. ? VN$(2));
- 'wà (watch) come '?', ma apre una sezione centrale nella finestra in cu numera da 1 a 7 le variabili e ne visualizza l'aggiornamento del valore ad ogni istruzione eseguita, es. wa VN$(2) -> 1) VN$(2)=...;
- 'wc' cancella numero watch oppure '*' tutti i watch, es. wc 1 o wc *;
- 'bp' (breakpoint) imposta fino a 9 punti in cui l'esecuzione del programma si interrompe e viene aperta la finestra di debugger. I punti sono specifica bili con l'etichetta seguita da il carattere ':' o con il numero di riga (es bp ETICHETTA: o bp 194); il brakpoint visualizza in reverse i numeri di riga ed è applicabile o cancellabile direttamente con [F8] sulla prima riga visualizzata nel debugger; con il tasto 'vai'[F5] si chiude il debugger per eseguire fino al breakpoint, Esc interrompe e riapre il debugger (riapribile anche con la riesecuzione del programma in caso di [F5] con assenza di breakpoint)
- 'bc' cancella 'bp', es. bc ETICHETTA:, bc 194, bc * ( * = tutti bp);
- 'bl' visualizza la lista di breakpoint (numeri riga o etichette);
- 'tl'visualizza la lista dei tasti in sospeso, in attesa di essere letti, inviati con PUTTAST.

# 8 – LINGUAGGIO SPRIX

Nella programmazione SPRIX la rappresentazione di un valore avviene tramite VARIABILE anche risultante da ESPRESSIONE. La variabile è utilizzabile nel momento in cui serve, senza doverla dichiarare in testa al programma e può rappresentare valore NUMERICO e ALFANUMERICO (STRINGA): il primo è un numero reale positivo o negativo con massimo 6 decimali separati da un punto senza arrotondamento della parte eccedente all'ultimo decimale stabilito

Esempio: A=2345.61, B=-12, NUM_01=-45.123456;

il secondo è un valore di massimo 255 caratteri fra doppi apici

Esempio: A$="Sprix", B$="AlFa:1", ALFA_1$=" Sprix 1"

La sintassi del nome della variabile determina quindi il tipo di valore che rappresenta: se termina con il simbolo '$' e ha il valore fra apici rappresenta una stringa altrimenti è un numero; il nome della variabile ha massimo 12 caratteri alfanumerici maiuscoli (A-Z e 0-9) e non può iniziare con numero o '_' e non può appartenere a nomi riservati a Funzioni, Istruzioni e parole chiavi

Esempio: VAL, STR$, END, GOTO, ALFA, NUM0,

L'utilizzo di variabile non valorizzata comporta una sua valorizzazione a zero (se numerica) o a nullo "" (se stringa); quella numerica alloca 8 byte di memoria e la stringa 4 byte più la lunghezza della variabile stessa.

La variabile può essere un ARRAY, cioè un vettore di uno o più valori in sequenza (ELEMENTI) dello stesso tipo (numerici o stringa), ciascuno dei quali è una variabile utilizzabile nelle espressioni come qualsiasi altra, ma accessibile direttamente o in modo ciclico tramite l'INDICE che rappresenta l'elemento. L'array consente fino a 3 dimensioni (matrice di vettori) da di chiarare con DIM seguito dal nome della variabile array con '()' in cui indicare numeri costanti interi non in forma di variabile, rappresentativi delle dimensioni e del numero di elementi di ciascuna dimensione, fino a un TOTALE MASSIMO di 262080 per l'array stringa e 131040 per il numerico, entrambi ricavati dal PRODOTTO degli elementi di ciascuna dimensione. Indici con valori fuori dalla dimensione dichiarata oppure elementi il cui prodotto eccede il consentito, provocano l'errore 'Indice dell'array fuori dai limiti'. Un numero di elementi non corretti o utilizzati prima del loro dimensionamento (anche quando richiamato in libreria), provocano errori 'dimensioni dell'array troppo grandi', 'array non dimensionato'. L'array non è gestito nella finestra parametri di esecuzione [F6] e non è azzerato nel campo di stampa [F2].

## 9 – SINTASSI FUNZIONI/ISTRUZIONI

Il dimensionamento dell'array va effettuato con un'apposita funzione che al pari di altre funzioni e istruzioni è governata da una determinata sintassi di parametri da rispettare per il corretto funzionamento; essi sono indicati con nomi da sostituire (con specifici valori o variabili) di tipo numerico o stringa se terminano con '$'; i simboli che racchiudono il parametro ne indicano la obbligatorietà '<...>' (anche quando è consentita l'indicazione di valori nulli ="" in caso di stringhe oppure zero =0 in caso di numeri) o la facoltatività [...]. Le parentesi tonde indicano la natura di funzione e devono essere sempre indicate. Esempio, dati i seguenti valori delle variabili A$="Ab12" oppure A$="" e N=68 oppure N=0 , la seguente ISTRUZ <par$>,[par] può essere e seguita nei seguenti modi:

ISTRUZ A$ o ISTRUZ "Ab12" o ISTRUZ A$,N o ISTRUZ "",68 o ISTRUZ "Ab12",0

## 10 – OPERATORI/ESPRESSIONI

L'assegnazione del valore a una VARIABILE può anche non essere costante, ma in forma di risultato di un'ESPRESSIONE (o anche di una STRUTTURA oppure un DIZIONARIO, trattati nella sezione degli archivi dei dati del gestionale).

L'ESPRESSIONE è un valore o il risultato di una combinazione di valori, costanti e/o variabili effettuata tramite OPERATORI (aritmetici, di relazione e logici) con la seguente priorità di esecuzione decrescente:

NOT -'negativò ^ * / MOD + -'sottrazionè < <= > >= = <> AND OR ( )

Gli OPERATORI ARITMETICI eseguono operazioni matematiche su valori numerici:

il meno unario di 'negativò con singolo operando
`es. -123`

e con priorità maggiore rispetto al meno di 'sottrazione' con più operandi
`es. 14-20 con risultato -6, C=20-6, D=A-2-1, ecc;`

elevazione a potenza '^' con esponente tra 1 e 32.767 e intervallo di risultato di +/- 999999999999
`es. 3^4 con risultato 81, E=10^2, F=E^2, ecc);`

moltiplicazione '*'
`es. 3*7 con risultato 21, C=10*2, D=B*2, ecc;`

divisione '/' (per zero è pari a +/- 999999999999
`es. 18/3 con risultato 6, A=10/2, B=A/2;`

resto della divisione 'MOD' con il valore degli operandi compresi nell'intervallo +/-2147483647 senza decimali
`es. 18 MOD 5 è il resto di 18/5 con risultato 3, G=100 MOD 3, H=11 MOD 2, I=E MOD 2;`

somma '+'
`es 2+3 con risultato 5, A=10+4+3, B=A+1`

che consente la CONCATENAZIONE di stringhe costanti e/o variabili fra esse unite
`esempio dato A$="Pass", B$="-Mexal" e CONCAT$=A$+B$+" 2009" il risultato di CONCAT$ è "Pass-Mexal 2009";`

infine le parentesi aperta '(' e chiusa ')' che consentono di invertire l'ordine di esecuzione che a parità di livello avviene da sinistra a destra così come si raggruppano automaticamente gli operatori eccetto quello unario (-123), di assegnazione e/o relazione (=) che si raggruppano da destra verso sinistra. Esempi di espressione e relativo equivalente con le parentesi:

```
saldo - capitale - interessi        ((saldo - capitale) - interessi)
saldo + -capitale + interessi       ((saldo + (-capitale)) + interessi)
tasse-10/100*saldo    ((tasse-10/100)*saldo)
primo = ultimo = 0    (primo = (ultimo = 0))
```

Gli OPERATORI DI RELAZIONE confrontano valori omogenei (numeri o stringhe) per ottenere un numero di tipo true (1) o false (0) su cui decidere tramite condizione IF ed in cui i confronti fra le stringhe seguono l'ordine alfabetico dove quella di ordine precedente è minore ("aa"<"b"), quella più corta nella sua intera misura è più piccola ("a "<"a  ") e infine, le minuscole sono > delle maiuscole che sono > dei numeri. Gli operatori di relazione sono: minore di '<'

```
(es. IF X>60 AND Y<20 THEN Z=X*Y);
```

minore uguale a '<='

```
(es. C=4: IF H<=3 THEN Q=10*3);
```

maggiore di '>'

```
(es. IF A>10 OR B<0 THEN X=A/2);
```

maggiore uguale a '>='

```
(es. Y=3: IF X>=3 THEN Z=10*3);
```

uguale a'=', utile anche per assegnare un valore a una variabile

Gli OPERATORI LOGICI sui numeri per collegare due o più relazioni in modo da decidere nello stesso modo degli operatori di relazione. Gli operatori logici sono 'non', 'è, 'oppurè: non 'NOT' (es. IF NOT (P=-1) THEN ...); e 'AND'; oppure 'OR'.

I valori combinati nell'espressione devono essere omogenei (numerici o alfanumerici) poichè caratterizzano il tipo di espressione (numerica o alfanumerica); diversamente non sono combinabili se non attraverso un loro trattamento con FUNZIONI per ottenere un certo risultato (es convertendo il numero in stringa o viceversa) e da questa regola non sono esclusi i valori data/tempo Es. espressioni N=6*(18+20): C$=A$+B$ dove A$="stringa 1 ": B$="stringa2"

```
IF <espressione> THEN <comando1> [ELSE <comando2>]
```

In Sprix/Collage esegue <comando1> se l'<espressione> è una condizione vera altrimenti esegue <comando2>. THEN è obbligatorio e ELSE è facoltativo, ma sempre tutto sulla stessa riga. La scorretta indentazione della IF dentro un ciclo (es. FOR) può rendere inoperativa la condizione.

Esempi:

```
- IF VA$="+" THEN TOT=IMPORTO1+IMPORTO2 ELSE TOT=IMPORTO1-IMPORTO2
- IF VA$="X" THEN IF B=0 THEN N=2 ELSE N=3
- IF VA$="X" THEN N=1 ELSE IF VA$="Y" THEN N=2 ELSE N=3
- IF N<1 THEN GOSUB SUB0 ELSE IF N<2 THEN GOSUB SUB1
```

# 11 – VIDEO E TASTIERA

# PARAMETRI DI ESECUZIONE - TASTO [F6]

In Sprix consente di definire fino a 10 variabili (no array) la cui valorizzazione viene chiesta in una finestra di input automatica all'esecuzione dello sprix che gestirà queste variabili. Nel campo 'Righe del modulò si può stabilire la quantità di righe di stampa del modulo (compatibilmente ad eventuali righe di stampa di tipo 'P'), altrimenti sono assunte quelle di configurazione del dispositivo di stampa scelto (il campo è obsoleto). I campi 'Nome' e 'Descrizione' della variabile sono di 12 e 20 caratteri; in ambienti non grafici la visualizzazione del secondo è completata con '....:' fino al raggiungimento della descrizione più lunga presente. Il campo 'Ln' stabilisce la quantità di caratteri e il campo 'Dec' il tipo di variabile: Numerica (inserendo il numero decimali da 0 a 6), Stringa (A o vuoto), Conto di Contabilità (C), Articolo di magazzino (M), Data in formato stringa (D) che richiede 'Ln' = 5, 8 o 10. Attiva le funzioni monitor del gestionale (ricerca, modifica, inserimento...). Vedere anche INPUT. La tabella non è importabile con [ShF7] e le variabili sono utilizzabili nel ciclo tabellare [F4]

# 12 – EDITOR - STAMPA

# TIPO RIGA IN STAMPA - TASTO [F5]

La pressione del tasto [F5] sull'EDITOR accede a sinistra dell'ordinata Y di separazione verticale. Con [F2] si apre un elenco per scegliere la tipologia di attributo di stampa alla riga corrente: 'A' di stampa in carattere automatico, cioè normale o compressa in base alla lunghezza di riga assegnata in sede di installazione alla stampante utilizzata (v. file di configurazione) secondo i seguenti rapporti fra Lunghezza riga stampante e Numero di caratteri della riga da stampare: Normale (80 contro <=79 e 132 contro <=131), Compresso (80 contro >79 e 132 contro >131). La riga 'A' è una riga 'N' che diventa 'C' se la riga da stampare supera la larghezza del modulo; 'N' di stampa in carattere normale; 'È di stampa in carattere espanso (non gestibile in 'Rtf'); 'G' di stampa in carattere grassetto; 'C' di stampa in carattere compresso (non gestibile in 'Rtf');'I' di stampa in carattere corsivo; 'F' di scrittura della riga come record su File esterno. Il nome file con eventuale percorso (tutto compreso fra doppi apici “c:\tmp\esempio.txt”) è obbligatorio e viene richiesto alla conferma di ciascuna riga, tranne nell’utilizzo della funzione generale [F5] di inserimento di nuova riga dello stesso tipo di quella su cui si trovava il cursore alla pressione del tasto con scorrimento verso il basso del testo. Il file viene occupato fino al termine del programma e solo dopo sono consentite operazioni di gestione file (open, copia,..); un'alternativa consiste nell'uso di stampante configurata di tipo File; 'P' invia alla stampante un salto pagina forzato; la riga non è scrivibile e la quantità di esse deve essere compatibile con lunghezza del modulo;'S' di stampa in carattere sottolineato (non gestibile in 'Pdf').

# CAMPO DI STAMPA - TASTO [F2]

La stampa dei valori di Variabili e di Dizionario sulla singola riga avviene tramite l'attivazione del campo di stampa con la funzione [F2] in cui definire nel 'Nome campò la Variabile o il Dizionario (ricercabile sempre con F2)e nei campi successivi le relative caratteristiche e attributi di stampa proposti e modificabili: la lunghezza di stampa (se riconosciuta c'è default con la misura, altrimenti è 15 caratteri); l'allineamento (default = automatico); il segno di valori negativi (default = a sinistra); il numero di decimali (default = solo quelli eventualmente presenti); il punto separatore di migliaia (default = assente); il valore a zero (default = non stampare); il carattere di riempimento per valori di lunghezza < alla dimensione del campo(default = blank); l'azzeramento delle variabili dopo la stampa, tranne per i valori di quelle utilizzate come indici di array (default = no). A seconda del tipo di valore che va stampato questo ha già una forma predefinita: es.data, ora, codice conto, codice tributo, protocolli Iva, codice articolo, anno gestito, anno in linea,ecc.

Il nome della Variabile da inserire nel Nome campo, può essere un valore numerico, alfanumerico e array, anche espressione o risultato di essa. Ad esempio: VARNUM, VARALF$, VN(3), VA$(2), ESPRESS=VARNUM+VAL(VARALF$), oppure una ESPRESS$=VARALF$+STR$(VARNUM,0,0). Alla conferma del campo il nome è trascritto in reverse nell'editor per la lunghezza prestabilita e prefissato con il carattere tilde (~).

## INTESTAZIONE DI STAMPA - TASTO [SHF5]

Con tasto [SHF5] si marca la riga corrente con 'H' che le attribuisce proprietà di intestazione di stampa ad ogni salto pagina. Non può essere riga di ciclotabellare, non può contenere ETICHETTE, GOTO e GOSUB). Consentite invece le istruzioni, assegnazioni e stampa del valore delle variabili in quel momento; consentiti gruppi di righe 'H' se contigue, non vuote e non precedute da altre righe stampa. Consentite più testate con memorizzazione dell'ultima incontrata per essere stampata automaticamente ad ogni salto pagina prima di stampare la prima riga.

## 13 – INTERATTIVITÀ

L'import/export di dati può avvenire in molti modi: con trasmissioni CVS o ASCII e altre modalità previste dallo standard del Gestionale se previste (come ad esempio l'Importazione Moduli del menu Servizi, che peraltro trova le corrispondenti Istruzioni IMOD-SPRIX), oppure tramite Sprix/Collage con utilizzo di FILE, RDBMS e possibilità di mirroring e esportazioni su database in chiaro. In questo modo è possibile interrogare e/o scrivere su supporti esterni. La possibilità di gestione delle porte di COMUNICAZIONE e effettuare SHELL di comando (e lancio Sprix) completano la interattività del Gestionale con il mondo esterno.

## 14 – DATABASE

ODBC: l'interfaccia ODBC (Open Database Connectivity) permette alle applicazioni di accedere ad un Database Relazionale (RDBMS) utilizzando il linguaggio standard SQL (Structured Query Language). Con funzioni di ODBC l'applicazione può accedere indifferentemente a Database per cui è stato sviluppato un driver ODBC senza dover essere modificata. L'implementazione dei comandi ODBC in Sprix/Collage adotta una sintassi simile a quella delle funzioni delle API ODBC di Microsoft, per usufruire della documentazione di ODBC disponibile. Per la creazione e collegamento del Database, vedere la documentazione del disco di rilascio nella directory "gest\doc\conces\". La chiave TipoODBC presente nel pxclient.ini può assumere i seguenti valori: 0 = DEFAULT (per MSWindows ODBC, per Linux JDBC); 1 = ODBC (MSWindows), UnixODBC (in LINUX);2 = JDBC (per MSWindows e LINUX). In seguito sono elencate istruzioni SQL di Sprix/Collage per l'accesso a RDBMS, file Xls, Csv,.ecc...

## 15 – COLLAGE

Il Collage esegue/blocca le routine di codice Sprix in base a eventi precodificati e che non siano relativi ad uno stesso programma Sprix. Collage dispone di specifiche funzioni, ma non effettua tutte le operazioni Sprix (accessi diretti in scrittura, letture tabellari, parametri di esecuzione.e quanto altro riservato a Sprix); Collage può eseguire uno Sprix (v. SHELLSPRIX).

Dalla voce Collage si accede allo stesso editor Sprix attraverso un elenco di programmi da installare e attivare in base a una sigla identificativa del percorso della voce di menu a cui i programmi devono fare riferimento per eseguire le loro routine in base ad eventi (ETICHETTE) stabiliti nel codice.

Collage non è eseguito come Sprix, ma viene caricato e eseguito all'accesso della voce di menu a cui è collegato. Premere [CTRL+G] per stampare le voci menu. Es. la voce 'CI' esegue routine in Contabilità-Immissione primanota.

Inserendo '*' il Collage è installato su tutte le voci del menu scelto (es. S* riguarda tutte le voci di Stampe) e sostituendo con 'AUTÒ è invece installato ovunque sia consentito e attivandosi al primo evento utile stabilito nel codice (occupa sempre della memoria, pertanto è bene non abusarne).

Il 'Nome Filè è modificabile [F4] e la sintassi predefinita alfabetica con lettere di selezione rapida menu (maiuscole o sottolineate) è personalizzabile (senza caratteri speciali non accettati dal sistema operativo, es.'*').

In quasi tutti i punti del gestionale viene effettuata la chiamata a routine Collage: se esiste viene eseguita e quando termina, riporta il controllo al gestionale (dopo l'eventuale modifica interattiva dello standard).

Più Collage sono installabili sullo stesso menu e se contengono la stessa etichetta evento le loro routine sono caricate contemporaneamente sullo stesso punto, con priorità d'esecuzione a quella del Collage che nell'elenco è dichiarato prima degli altri Collage. A parità d'etichetta su Collage diversi il menu specifico ha precedenza rispetto al Collage AUTO, anche se gerarchicamente di ordine inferiore nell'elenco dei Collage; l'ordine è importante e si rende disponibile una funzione di cambio di posizione fra i Collage [ShF6] e di attivazione e disattivazione del Collage [F6], visibile con il flag '-'. Mantenendo la coerenza con le etichette nel programma, il Menu è successivamente modificabile [F2]. A differenza dello [ShF3] lo [ShF6] sulla stessa posizione cancella il formato conservando il sorgente nella directory

Quando sono caricati i Collage, la sequenza di Aiuto [F1+F1] consente di selezionare i pulsanti Collage [F2] e Etichette [F3]: il primo visualizza il motivo di attivazione di ciascun programma (da menu, da etichetta, da altro Collage), il secondo lo stato del Collage attivo che rispetto alla specifica etichetta collegata allo specifico campo di input è considerato in memoria o ancora da caricare o disattivato da una specifica istruzione. Più Collage contemporaneamente attivi sulla stessa sessione di più terminali in rete, richiede opportuna gestione di semafori (blocchi) in caso di accessi ad una unica risorsa (File, Com, ecc.) i file dei Collage attivi sullo stesso menu sono letti dall'alto al basso: l'accesso allo stesso dato comporta che, se in un determinato evento fosse eseguita una determinata operazione nel primo file, l'operazione eseguita precedentemente nel medesimo evento potrebbe essere persa nel successivo file.

# 16 – EVENTI COLLAGE

Ciascuna routine Collage va inizializzata con un'ETICHETTA e terminata con END. Alla chiamata della routine alcune variabili di sistema sono automaticamente settate (_SINP$,_TF,_NCAM$,_NFIN,_DES612$,...). La sintassi dell'etichetta per l'esecuzione della relativa routine al verificarsi dell'evento associato all'etichetta, si forma con le seguenti modalità: PREDEFINITE, PERSONALIZZABILI e per le STAMPE. Quelle predefinite sono:

**ON_START:**

**ON_END:**

# 17 – EVENTI COLLAGE SU ARCHIVI

ON_START??:(all'ingresso), ON_PUT??:(alla scrittura), ON_DEL??: (alla cancellazione), ON_END??: (all'uscita), dei moduli indicati in sostituzione del carattere '??'

| | |
|---|---|
| AR | anagrafica articoli, |
| BL | bolla lavoro |
| CFR | Referenti |
| CONT | anagrafica contatti. |
| INDSPED | indirizzi di spedizione |
| MM | emissione documenti magazzino |
| PC | piano dei conti |
| PN | prima nota |
| TZ | particolarità |

Per esempio il modulo PC ha gli eventi ON_STARTPC:, ON_PUTPC:, ON_DELPC:, ON_ENDPC:.

I tasti che attivano l'evento possono variare a seconda del modulo e delle circostanze dell'operazione; per esempio, l'uscita può essere effettuata con i tasti [Esc],[Fine],[Invio],[F10]... (ma non con [ShF10]).

NB: l'evento predefinito potrebbe non manifestarsi necessariamente da tutti i punti del gestionale che ne consentono l'operazione associata. Per fare alcuni esempi: da menu magazzino 'MD' l'evento ON_DELPN non si manifesta al

ON_DELMM; l'evento ON_PUTMM nel modulo “ME” si attiva solo con l'effettiva scrittura, ad esempio la trasformazione e quindi, non in ristampa (in questo caso è attivo l'ON_STARTMM); ; ON_DELINDSPED nel menu 'MI' poichè c'è la sola cancellazione logica (che è un PUT di trasformazione). Alcuni eventi richiedono l'uscita dal primo campo di input, per esempio: ON_STARTPC in 'PC' e ON_STARTBL in 'PB'.

Generalmente negli eventi PUT e DEL la semaforica è attiva ed il record è fisicamente già aggiornato, mentre in END la semaforica sul record è già disattivata; queste regole non sono universali e vanno verificate: esempio, ON_ENDMM in emissione di magazzino scatta prima dello sblocco semaforica sul documento e quindi, altre operazio ni lanciate al ON_ENDMM sullo stesso documento trovano blocco input tastiera.

Altri eventi predefiniti sono: ON_BEEP: (all'emissione di segnale acustico di avviso definito nel file .ini della componente desk); ON_LISTALOTTI: (prima di scrivere il documento, nuovo o riscritto con ulteriori righe nuove con

autoprelievo attivo, rende disponibili gli identificativi lotti di autoprelievo per la riga che si sta processando).

La variabile Collage _NCAM$ contiene il CODICE del campo in cui si trova il cursore in quel momento e che è utile alla creazione di etichette. Questo CODICE è visibile anche nella finestra aperta con [F1F1] sul campo e che ne visualizza i possibili Eventi Collage, ed es. ON_IN_CODICE: (all'ingresso), ON_OUT_CODICE: (all'uscita), ON_ERR_CODICE: (alla visualizzazione di errore) e ON_612_CODICE: (alla pressione del tasto [F12]).

Con la variabile Collage _DES612$ si può stabilire al momento opportuno (ad esempio al ON_IN_CODICE: del medesimo campo) la descrizione del tasto [F12] e se più Collage sono caricati contemporaneamente in memoria dalla stessa etichetta [F12], la stessa _DES612$ visualizza le descrizioni per scegliere quale routine eseguire); esempio: due Collage entrambi su 'MA' (modulo magazzino anagrafica articoli)

```
ON_IN_MADSART:                   |ON_IN_MACODAL:
  _DES612$="primo F12": END      |   _DES612$="secondo F12": END
ON_612_MACODAL:                  |ON_612_MACODAL:
  VIMSG 4, "routine 1": END      |    VIMSG 4, "routine 2": END
```

ATTENZIONE: se il CODICE del campo non c'è o non è in maiuscolo o non è relativo a un input, l'evento può essere riservato o non esistere o infine vincolato (es. nessun azienda confermata); sostituendo il CODICE con la parola 'ALL' (tutti i campi) si verifica se è consentito questo metodo alternativo: ON_IN_ALL:, ON_OUT_ALL:, ON_ERR_ALL:, ON_612_ALL:. Debuggare per verificare sequenze corrette o ridondanze cicliche o per sfruttare eventi prece denti/successivi a quelli non utilizzabili; le sequenze degli eventi dipendono anche dal tipo di modulo e eventuali opzioni/parametrizzazioni possibili.

Alcuni moduli di particolare importanza come 'ZCC' (Utenti e chiavi d'accesso) non consentono di sviluppare Collage e le ragioni sono intuibili.

# 18 – EVENTI DELLE STAMPE

Gli elementi per la costruzione dell'etichetta di stampa si ricavano premendo il tasto [Ctrl+G], nella finestra mostrata a video scegliere "Impostazione stampa righello" che richiede nella stampa mostre il righello (informazioni di impaginazione); ciò vale anche nelle stampe FISCALI grafiche (ma se definitive, le etichette non sono attive). Gli elementi sono il CODSTAMPA (Codice stampa) e CODRIGA (codice di riga) ad esempio, nel modulo del documento il Codice stampa si chiama 'MODULÒ e i vari codici di riga si chiamano TESnn...CORnn...PIEnn...

Indicando il singolo CODSTAMPA la routine è eseguita solo in quella stampa ed indicando il singolo CODRIGA è eseguita solo in quella riga, ma se uno dei due (e/o entrambi) sono sostituiti dalla parola 'ALL' l'esecuzione è generalizzata (tutte le stampe e/o tutte le righe). ALL è utile se non esistono dei CODSTAMPA e CODRIGA (che vengono ricercati prima, indifferentemente dalla loro posizione all'interno del collage; ma non tutte le righe (anche con codice) sono intercettabili, specie se di testata/piede, in quanto riservate e di uso esclusivo (e in genere hanno prefisso '_'); le utilizzabili sono quelle del corpo. Le etichette eventi sono le seguenti:

ON_STARTST_CODSTAMPA:

alla partenza di stampa dopo l'apertura stampante

ON_ENDST_CODSTAMPA:

alla fine della stampa, prima di chiusura stampante

ON_ST_CODSTAMPA_CODRIGA:

prima di stampare la riga su stampa e rende disponibile in _SST$ la stringa da stampare - es. per modificarla - e terminata la routine, la stampa. Esempi: ON_ST_MODULO_ALL: _SST$="variariga" : END;
ON_ST_BILANCIO_RIGA: VIMSG 4, _SST$ : END;

ON_TST_CODSTAMPA_CODRIGA: prima della stampa rende disponibile nell'array_TSST$(1-99) la stringa da stampare; modificando _TSST$(1) si modifica la riga e tutti gli elementi aggiunti - dal 2 compreso in poi -, sono considerate nuove righe in stampa dalla posizione corrente; azzerando tutti gli elementi(= "") la riga viene soppressa fisicamente compattando la stampa. L'aggiunta di righe in modulistica oltre il numero stabilito nel MODULO disallinea la pagina - il conteggio è successivo alla stampa -. Le stampanti di tipo 'S' consentono un dfs che annulla il saltopagina (diversamente modificarlo impostando il "comando spg:no;").

Negli eventi ON_TST_ di testata "TESnn" e di piede "PIEnn", l'aggiunta di righe, annulla l'evento collegato al numero di riga immesso: es. se all'evento"ON_TST_MODULO_TES01:" si caricassero _TSST$(1) e _TSST$(2), l'evento successivamente intercettabile non sarebbe intuitivamente "ON_TST_MODULO_TES02:",bensi' "ON_TST_MODULO_TES03:".

_SST$ e _TSST$(1-99) sono variabili globali (attenzione con routine contemporanee a altri Collage) e funzionano con carattere a spaziatura fissa; non sono attive nel utilizzo di variabili modulistica VTXT/VTDE (campi riservati a stampanti di tipo 'W' e/o "PDF" e quindi già processati del gestionale). La_TSST$() non è adatta a modelli prefincati e non è attivabile nelle stampe fiscali contabili (anche se prevedono un CODSTAMPA) data l'incompatibilità con la numerazione automatica del documento bollato.

L'evento di stampa non anticipa quello di default del gestionale e può essere attivato anche da alcune condizioni (es. anno di ingresso nel programma <2002 per il modulo 'SFD', tipo di scadenzario, tipo dell'azienda per moduli SFG, SRA, SRV, NEC,...).

L'attivazione di alcuni codici riga, dipende dalle condizioni in cui vengono eseguite le relative stampe; ad esempio, nella stampa dell'estratto conto normale, se nelle condizioni commerciali della anagrafica Cli/For esiste il conto dell'agente, saranno stampate entrambe le righe INT1 e INT2,altrimenti sarà stampata solo INT2; in alcune stampe l'attivazione d'una stampa estesa anzichè ridotta, oppure l'attivazione di determinati flag,rende disponibili alcuni CODRIGA che precedentemente non lo erano.

Per le stampe personalizzate è attiva una sola label "ON_STP_DOC:".

Le uniche stampe personalizzate su cui è attivo il collage sono:

Magazzino "Movimenti di Magazzino"

"Documenti di Magazzino"

Attivata la label nel sorgente collage la lettura delle informazioni per le righe elaborate avviene tramite l'esecuzione del comando "GETATTMM".

Le variabili da valorizzare sono:

_STPGXA$(1-20) : _STPGXN(1-20)

Variabili generali, valorizzabili e modificabili sempre. Definiscono un valore che sarà riportato alla fine dell'elaborazione nella relativa variabile di stampa.

_STPTXA$(1-20) : _STPTXN(1-20)

Variabili di documento, attive per ogni riga elaborata dalla stampa.

_STPRXA$(X,1-10) : _STPRXN(X,1-10)

Variabili bidimensionali (X = riga documento, elemento vettore).

Attive per le righe del documento. L'utilizzo delle variabili è identico a quello delle variabili collage _MMSWA$(_NMM,1-10) e _MMSWN(_NMM,1-10), a cui si rimanda per ulteriori dettagli.

```
Esempio:
ON_STP_DOC:
 _STPTNO = 0
 GETATTMM
 IF _MMSIG$ = "FT" THEN _STPTXA$(1) = "Valore personalizzato"
 FOR I = 1 TO _NMM
  IF _MMSIG$ = "FT" THEN _STPRXA$(I,1) = "Valore personalizzato"
  NEXT
  IF _MMSIG$ = "BD" THEN _STPTNO = 1
END
```

Nell'esempio, il collage, leggendo i documenti, in caso la sigla sia "FT" imposta un valore nelle variabili dichiarate, per le "BD" esclude la riga elaborata dalla stampa.

## 19 – ARCHIVI DEL GESTIONALE

In aggiunta alle variabili personali dell'utente, lo Sprix dispone di specifiche variabili collegate a operazioni (funzioni e istruzioni) sugli archivi del Gestionale. Queste variabili si definiscono di STRUTTURA e di DIZIONARIO e richiedono una corretta sintassi e dei precisi parametri di gestione.

## 20 – VARIABILI DI STRUTTURA

Le variabili di STRUTTURA consentono di operare negli archivi del gestionale con istruzioni di lettura (GET?), scrittura (PUT?) e cancellazione (DEL?) e che generalmente terminano con la sigla dell'archivio coinvolto (es. GETPC legge il piano dei conti). Le variabili di struttura hanno sintassi _XX?$():

_XX è l'archivio coinvolto (es. '_PC' archivio piano dei conti);

? è il dato dell'archivio (es. 'PAG' codice pagamento nell'archivio PC);

$ è l'eventuale stringa (es. 'COD$' codice conto nell'archivio PC);

() è l'eventuale array (es. 'NDS$(1-2)' descriz. delle 2 note del conto);

Dall'esempio si ottengono cosi' le variabili _PCPAG, _PCCOD$, _PCNDS$(I),.tutte relative all'archivio del piano dei conti da leggere (GETPC), scrivere (PUTPC), cancellare (DELPC) e controllarne gli esiti con la _ERRPC; tutte le operazioni possibili (non solo GET,PUT,DEL) sono infatti controllate dal gestionale e segnalate con variabili di sistema _ERR? che segnalano l'eventuale errore e non sono forzabili e nè aggirabili. Non sempre questi controlli sono attuati dal gestionale, poichè dipende da che cosa è attivo (per es. il controllo di congruità della primanota avviene solo con scadenzario attivo)

Le variabili di struttura sono elencate nella sezione degli archivi del Gestionale nelle corrispondenti istruzioni e funzioni che le governano; in funzione della evoluzione del Gestionale, alcune nascono 'rinnovatè nel nome e nella sostanza affiancandosi nell'elenco a quelle originarie, per consentire di allineare e attualizzare l'eventuale revisione del vecchio sorgente a versioni recenti del gestionale. L'elenco di variabili non riporta tutte le informazioni sulla gestione dei dati coinvolti, pertanto occorre consultare il manuale utente. Alcuni archivi del gestionale, es. le tabelle, non hanno variabili di struttura e sono modificabili solo dal gestionale o con Collage.

# 21 – VARIABILI DI DIZIONARIO

Le variabili di DIZIONARIO consentono la sola lettura degli archivi dell'azienda aperta e dell'applicazione in generale. Sono quasi omonime della struttura, ma vanno scritte in minuscolo e non sempre per una struttura esiste il corrispondente dizionario o viceversa. Il dizionario si consulta dalla voce 'stampa Dizionariò nel sottomenu 'Personalizzazioni-Sprix/Collagè del menu Servizi' oppure anche in modo guidato direttamente dall'editor con il tasto[F2] che apre un elenco di gruppi di archivi scorribili e selezionabili con Invio per accedere a sottoelenchi di tabelle archivi contenenti altri sotto elenchi di nomi di variabili di Dizionario da selezionare con [Invio].

In testa alla tabella possono esserci parametri corrispondenti alle prime voci del dizionario e che rappresentano le CHIAVI DI LETTURA per identificare quelle successive (ulteriori indicazioni sulle chiavi di lettura sono consultabili nella directory "gest\doc\conces\dbmirror\tabelle.htm" del disco della versione). Per esempio il prezzo di listino di un articolo 'arprz' richiede l'indicazione di quale listino (I) e quale articolo arcod(), di conseguenza la relativa sintassi è arprz(I)(arcod()) in cui sia I (l'indice d'array) che arcod() sono indicabili direttamente con un determinato valore, es. I=2,

COD_ART$ = "CODICEARTICOLO" e quindi arprz(2)("CODICEARTICOLO"), o indirettamente con un'altra variabile, anche di Dizionario o di Struttura, ad esempio arprz(2)(COD_ART$) o arprz(2)(afcod()) o arprz(2)(_ARCOD$). L'importante è che riportino comunque un valore equivalente al parametro richiesto. Si possono fare altri esempi come la prima nota per leggere la descrizione del piano dei conti pcdes(pncto()), la lettura taglie mmtag(I)() mtdes(mmntt(),I),

oltag(I)() o la lettura di scadenze pspim(I)() e cosi' via, fino ad arrivare alla necessità di sequenze di parametri complesse come sono richieste dalla quantità di inventario aqinv(arcod(),1,sxann) o dalle particolarità prezzo dell'articolo tzprz(1)("P","",1,arcod(),1,sxdat).

A destra di ciascuna voce c'è la relativa lunghezza e la natura numerica o alfanumerica della variabile, con eventuale numero di elementi array fra parentesi e da indicare prima dei parametri; es. il prezzo del secondo listino dell'articolo "ART1" nella tabella 'ar' di anagrafica articoli è dato da arprz(2)("ART1"). Per alcune variabili è attivo l'Help in linea [F1].

# 22 – CICLOTABELLARE (TABELLE DIZIONARIO)

Se la tabella è indicata 'senza parametri' le voci sono a lettura diretta in quanto uniche. Ad esempio, il dato di sistema relativo all'anno di gestione 'sxang' oppure all'anno in linea 'sxann'. Se l'indicazione 'senza parametri' è evidenziata la lettura dei record della tabella è solo sequenziale attraverso il CICLOTABELLARE (non eseguibile in Collage) che apre l'archivio

assumendo automaticamente i parametri della tabella (e che quindi non sono necessari), lo legge e lo chiude al termine del programma. Il ciclo tabellare viene battezzato con la sigla della tabella: si attiva nell'editor con [F4] e si cancella sulla stessa riga di attivazione con [ShF3]; nel primo campo Cod.Tab' con [F2] si cerca la tabella di interesse da confermare con Invio,nei campi successivi non obbligatori si selezionano le variabili Dizionario [F2+Invio+Invio]; nel campo di 'Sceltè si può inserire un'espressione logica fino a 220 caratteri per condizionare la scelta dei record, anche tramite valori di Dizionario prestabiliti fuori dal ciclo, per esempio posta la variabile alfanumerica ART$="ART1" (codice articolo), la scelta arcod()=ART$ seleziona solo "ART1" o la scelta mmsig()="FT" AND mmart()=ART$ seleziona nei movimenti solo fatture con articolo "ART1", oppure ppsaf(pccod(),sxann)>=0) seleziona i saldi positivi dei conti dell'anno in linea, e cosi' via..

Ciascun campo 'Ordinè di massimo 66 caratteri, consente espressioni fino a nove variabili e/o porzioni di esse, concatenate con '+', per creare gruppi gerarchici decrescenti di ordini di lettura (da 3 a 1) ed ottenere una vista logica di estrapolazione dei record. Senza ordinamenti la lettura è ordinata per chiave primaria del archivio, più rapida di ordinamenti articolati o

che ripetono la stessa chiave (ad esempio, inserendo nel ciclo di lettura di documenti di magazzino un ordine 3 con sigla+serie+numero+conto del documento o in modo equivalente la sigla su ordine 3, la serie sul 2 e il numero su1). Se l'ordine di lettura non è necessario, non utilizzare l'ordine.

Confermando il ciclotabellare con [F10] si marca la riga dell'editor con la sigla della tabella scelta visualizzata in reverse a sinistra della riga di separazione verticale. Il ciclo è estensibile ad altre righe contigue premendo F4 su di esse, per creare gruppi di istruzioni e/o di campi di stampa da ripetere tante volte per quanti sono i record letti. Ciascuna riga marcata può essere definita una rottura di livello dell'ordinamento con [ShF4].

In ciascun livello è infatti gestita la relativa rottura per condizionare la lettura con opportuni indicatori, che vengono accesi e eseguiti quando si verificano le condizioni presenti nell'espressione indicata nel gruppo gerarchico, per effettuare stampe di totali, salti pagina, calcoli, ecc... nelle righe in cui questi indicatori sono opportunamente posizionati. I livelli di rottura di ordinamento sono quindi tre (3, 2, 1), con due tipi di indicatori ciascuno, di intestazione 'I' (rispetto al record precedente) e totale 'T' (rispetto al record successivo): I3 e T3, I2 e T2, I1 e T1, evidenziati nel editor in reverse a fianco della sigla del ciclotabellare. Nella sequenza di righe di ciclotabellare l'indentazione delle colonne nell'editor deve essere decrescente da sinistra verso destra per le 'I' e crescente da destra verso sinistra per le 'T'. La sequenza dell'indentazione non si può invertire.

Sul primo record letto sono rotti tutti i livelli di ordinamento ed attivati tutti gli indicatori 'I'. Se nel record letto c'è la rottura di ordinamento rispetto al record precedente sono attivati gli 'I', altrimenti se c'è rottura rispetto a quello successivo sono attivati i 'T'. Per ciascuna rottura di livello superiore si attivano automaticamente le rotture di tutti i livelli inferiori dello stesso tipo (I3,I2,I1)->(I2,I1)->(I1) e (T3,T2,T1)->(T2, T1)->(T1). Sul ultimo record letto sono rotti tutti i livelli d'ordinamento con l'attivazione dei 'T'. Esempio: posto un insieme non ordinato di record di clienti, per contarli e totalizzarli per gruppi gerarchici Comune, Provincia e Regione si cicla la tabella dei clienti con Ordine 3: Regione, Ordine

2: Provincia e Ordine 1: Comune. Confermato il ciclotabellare (eventualmente preceduto da righe di intestazione) l'estensione nel editor di righe marcate come ciclotabellare (es. num. 8 righe), consente di stabilire quanto segue:

1) (R=1) riga capo del ciclo con rottura I3 per settare il contatore Regione(impostato fuori ciclo non verrebbe inizializzato ad ogni record) (a seconda dei casi può differire di +/-1 rispetto ai record)

2) (P=1)  riga con rottura I2 per settare il contatore Provincia

3) (C=1)  riga con rottura I1 per settare il contatore Comune

4) [~pc???] riga di campi stampa di variabili d'interesse lette

5) (R=R+1:P=P+1:C=C+1) riga di incremento contatori per record letto; può contenere anche altri calcoli, totalizzazioni, decrementi (X=X-1) e condizioni IF ... THEN Y=Y+1, ecc...con risultato da riportare poi in stampa alla rottura

6) [~C-1] riga di stampa con rottura T1 del totale per Comune

7) [~P-1] riga di stampa con rottura T2 del totale per Provincia

8) [~R-1] riga di stampa con rottura T3 del totale per Regione

Con EXITTAB si forza l'uscita dal ciclo tabellare (ad esempio quando si verifica una condizione) per proseguire con il codice successivo al ciclo. Sono consentiti fino a 99 ciclitabellari per programma e per evitare letture falsate è SCONSIGLIATO inserire nel ciclotabellare le operazioni GET-PUT-DEL in quanto è preferibile adottare altri metodi di lettura ciclica su array in memoria o altro supporto (es. file) precaricati con ciclotabellare.

Il ciclotabellare legge l'anno di apertura azienda. Elaborazioni di più anni in linea di archivi non multiannuali, avvalersi di APRIAZIE per effettuare singole aperture azienda su ciascun anno da ciclare, esportandone i risultati per effettuare le elaborazioni globali (confronti in memoria di valori di cicli tabellari operanti su anni diversi).

I ciclitabellari non sono nidificabili e per estrapolare variabili non appartenenti alla tabella ciclata, occorre indicare nelle stesse i precisi parametri necessari alla lettura. L'indicazione può essere diretta, anche tramite un'altra variabile, come può avvenire anche fuori del ciclotabellare, esempio, dato il ciclotabellare 'pn' sulla primanota e data la variabile conto C$="201.00001", la variabile 'descrizione del contò pcdes() può essere indicata pcdes(C$) o pcdes("201.00001"); l'indicazione può essere anche con ciclotabellare, esempio, dai movimenti di magazzino 'mm' per leggere i parametri necessari alle variabili del piano dei conti pccog(mmcli()) e dell'anagrafica articoli ardes(mmart()) oppure dalla primanota 'pn' alla descrizione del piano dei conti pcdes(pncto()); il caricamento dei parametri può avvenire su array con ciclotabellare su 'ps', esempio ARRAY(I)=psnum(), da ciclare su 'pn' dove scegliere ad esempio, pnndo()=ARRAY di tanti elementi per quanti sono stati caricati col primo ciclo.

Le necessità di ordinamento dei dati quando non è consentito dal ciclotabellare sono soddifatte con export dei dati (file, array..) o funzioni SORT.

L'uso del dizionario richiede quindi un'accurata conoscenza del Gestionale e eventuali indicazioni particolari nell'uso degli archivi come:

- sxpas si legge se il parametro SPRIXPW="S" nelle Chiavi di accesso;
- sxcso/sxcat dell'azienda "G" (gestione sottoazienda) è = al codice "651";
- pcage(pccod())="" significa assenza di associazioni nelle condiz. comm.li;
- pnord() e mmord() sono criteri di ordinamento righe progressivi (3,4,7,..)e non necessariamente in sequenza (1,2,3,4,5...); possono essere riordinati in memoria, per ottenere la stessa sequenza visualizzata a video;
- psnpi() e psdar() riportano valori solo per documenti > al 2001 (anno della loro implementazione); lo scadenzario ha 250 rate, 12 contropartite sospese e 24 rate professionisti; nelle provvigioni 'ps'-'pd' il maturato va calcolato (non è memorizzato);
- ordinamenti tabelle 'cs','ck','cz': cstab,cscto,cscod,csann,cstip,csdti -ckcto,ckann,ckccr,ckdat,ckprg - czann,cztip,czcto,czccr (la riga di tipo 'A' è la sommatoria di tutte le righe 'L' - suddivisione per Conto e per Centro di costo/ricavo della spesa sostenuta -). Il valore totale del cespite è la sommatoria delle righe comuni a cscod(),cstab(),cscto(),csann()
- mmtds() e mmdsc() e altre variabili ?tds() e ?dsc() relative alla lettura di righe descrittive dei documenti magazzino di tipo punto(.) e virgola(,) sono attivate in Sprix con LEGGIDESC <par> (se <par>=1 sono abilitate, il default è <par>=0 cioè disabilitate) per leggere il secondo record fisico del corpo documento e quindi, non ordinabile sulla stessa linea del record principale in caso di letture sequenziali (ma solo sulla successiva);
- fatturato totale del documento magazzino è un dato calcolato (non esiste)e occorre sfruttare le funzioni di riga del dizionario o letture a video;
- la tabella ordini 'or' riguarda OC, OF, PR
- odtip() è condizionato dal parametro "Sost.Autom.ord.matrice";
- bolle/fatture ordinate per indirizzo di spedizione: questi documenti sono ordinate per numero progressivo sezionale e le bolle in fattura per data documento e codice indirizzo spedizione (partendo dalle null ""). Per ottenere la stessa

numerazione disattivare "raggruppa documenti". Alternativa: statistiche movimenti articolo (non per indirizzo); ciclotabellare movimenti ordinati per mmspe()

- in bolla di lavoro 'ob' se non è specificato l'ordine, l'ordinamento è per obnum(),obnso(). La prima sottobolla di ogni bolla è la numero -1 bolla padre, fittizia e creata in automatico da programma. Nella tabella 'ol'se l'ordine non è specificato, l'ordinamento è olnum(),ollav(),olart().

Nota: ollav() utilizza il valore 32000 quando fa riferimento alla BL principale e di conseguenza -32000=Pronto, 0=PF sospeso, 32000=in lavorazione.

Per trattare tutte le righe di una determinata BL, selezionare tutte le righe con campo ollav in comune (sia positivo che negativo). In tabella 'oi' l'assenza di ordine genera ordinamento per oinum,oisot,oirif,oifas,oimag.

Esempio: documenti trasformati e contenenti descrizioni nel corpo

```
       |LEGGIDESC 1
       |+------------------------------------------
       ||Cod.Tab.:mm  MOVIMENTI DI MAGAZZINO - TESTATA DOCUMENTO
       ||Ordine 3:mmtsi()+mmtse()+mmtnu()
       ||       2:mmosi()+mmose()+mmonu()
       ||       1:mmord()+mmart()+mmdsc()
       ||Scelte..:
       ||         mmtsi()="BC" OR mmtsi()="FT"
              mm I    |
      mm   I N|BC/FT   ~mmtnu
      mm   I N|-------------
mm     N|OC      ~mmonu
mm     N|Righe   ~mmart()\\ Descr. ~mmdsc()\\\\\\\ Qta ~mmqt   Prez
~mmprz()\\
mm      |TOT=TOT+mmprz()
mm   T N|----------------------------------------------- Tot Prez
~TOT\\\\\\
mm   T  |TOT=0
```

## 23 – IMOD-SPRIX (Import Moduli in Sprix)

Complesso di istruzioni e variabili Sprix per importare dati da file ASCII mediante definizioni di aree testo oggetto di import; l'insieme delle definizioni di aree compone il formato del modulo per l'esecuzione dell'import. La definizione guidata delle aree testo per generare il formato è consentita tramite la funzione video [SH+F6] dell'editor Sprix (apre due voci fra cui scegliere quella dal nome " EDITOR MODULISTICA "). In Sprix si devono indicare le caratteristiche del documento e le aree da cui prelevare i dati ed i campi d'assegnazione; al documento va indicato il percorso (anche standard) ed il nome del file da importare.

Dal menu Servizi\Personalizzazioni tramite "Importazione Moduli" si possono generare programmi SPRIX che si aggiungono automaticamente a quelli gia installati dall'utente (per chi non volesse definire e strutturare il modulo e la generazione del sorgente Sprix relativo attraverso l'editor Sprix). La funzione di Importazione Moduli prevede la voce Tipi sostituzioni per stabilire una tabella di correlazione fra valori dove fosse necessario; anche in Sprix è prevista questa funzione, passando dall'EDITOR MODULISTICA [SH+F6] e premendo [SH+F3] sull'area selezionata. Per approfondimenti si rimanda alla relativa sezione del manuale utente.

## 24 – MYDB

Di seguito tutte le funzioni istruzioni variabili di Mydb

### Istruzioni

| | |
|---|---|
| MYDBTAB | Imposta la tabella per il ciclo tabellare |
| MYDBGET | Lettura di un'anagrafica |
| MYDBPUT | Scrittura di un'anagrafica |
| MYDBDEL | Cancellazione di un'anagrafica |
| MYDBPREV | Si posiziona sull'anagrafica precedente |
| MYDBNEXT | Si posiziona sull'anagrafica successiva |
| MYDBVIEW | Finestra di ricerca di un archivio |
| MYDBINPUT | Input a finestra di un'anagrafica |
| MYDBGETATT | Lettura dell'anagrafica corrente (Solo COLLAGE) |
| MYDBPUTATT | Scrittura a video dell'anagrafica corrente (Solo COLLAGE) |
| 35 | Indice da utilizzare con AZZVARSYS per azzerare le variabili di MyDB |

### Funzioni

| | |
|---|---|
| MYDBLOF | Ritorna il numero di chiavi inserite |

### Variabili

| | |
|---|---|
| _MYDB$() | Dati dell'archivio |
| _MYDBELE | Numero di elementi MYDB$() impostati |
| _MYDBETIC$() | Etichette dei campi MYDB$() impostati |
| _ERRMYDB | Valore numerico errore istruzione MyDB |

| | |
|---|---|
| _ERRMYDB$ | Messaggio errore istruzione MyDB |
| _MYDBK$() | 5 elementi per la chiave MyDB (Utilizzato solo il primo) |
| _MYDBID | Numero ID anagrafica corrente |
| _MYDBNR | Numero record anagrafica corrente |
| _MYDBARC$ | Codice archivio corrente (Produttore Archivio) Valido solo per MYDBGETATT, MYDBPUTATT |
| _MYDBANN$ | "S" se l'anagrafica è annullata logicamente |
| _MYDBCOD$() | Codice campo per i vari elementi _MYDB$() |
| _MYDBBLOC$ | "S" se l'anagrafica è bloccata da Passepartout (non modificabile) |

### Label

| | |
|---|---|
| ON_STARTMYDB: | Richiamata all'ingresso di un'anagrafica |
| ON_STARTFILTROMYDB: | Richiamata all'ingresso del filtro di un'anagrafica |
| ON_CTRLMYDB: | Richiamata all'uscita di un'anagrafica |
| ON_CTRLFILTROMYDB: | Richiamata all'uscita del filtro di un'anagrafica |
| ON_ENDMYDB: | Richiamata all'uscita di un'anagrafica |
| ON_ENDFILTROMYDB: | Richiamata all'uscita del filtro di un'anagrafica |
| ON_PUTMYDB | Richiamata dopo la scrittura di un'anagrafica (e dopo ON_ENDMYDB) |
| ON_DELMYDB: | Richiamata dopo la cancellazione di un'anagrafica (e dopo ON_ENDMYDB) |
| ON_STARTMYDBRIC: | Richiamata all'ingresso di una ricerca di un archivio |
| ON_CTRLMYDB: | Richiamata all'uscita di un'anagrafica |
| ON_CTRLFILTROMYDB: | Richiamata all'uscita del filtro di un'anagrafica |

ON_CTRLFILTROMYDB e ON_CTRMYDB sono richiamate dopo i controlli del programma ma prima del ON_ENDMYDB:, sono attivi sia MYDBGETATT che MYDBPUTATT e la variabile _TF che contiene il codice del tasto di uscita. Se si modifica _TF o si esegue la MYDBPUTATT, il programma rientra in anagrafica, per visualizzare le modifiche e rieseguire i controlli sull'anagrafica. Se si imposta _TF con un tasto valido questo viene elaborato.

### Dizionario

| | |
|---|---|
| sxmyd | Nome archivio MyDB principale in revisione |
| sxarc | Nome archivio MyDB corrente (cioè usato da <sxmyd>) Revisione di un'anagrafica all'interno di un'altra anagrafica |

ydPRODUTTORE_ARCHIVIO    yd + 10 caratteri del produttore + _ + 8 caratteri del nome archivio

## _AB

variabili di struttura archivi Codici Articoli Abbinati (GETABBIN-PUTABBIN-DELABBIN)

| | |
|---|---|
| _ABPRO() | (Numero progressivo; Specifico = 0,1,2,..; AUTO=-1) |
| _ABCOD$() | (Codice abbinato) |
| _ABCAT$() | (Categoria abbinamento) |
| _ABCATD$() | (Descrizione categoria) |
| _ABART$() | (Codice articolo) |
| _ABNO1$() | (Nota 1) |
| _ABNO2$() | (Nota 2) |
| _ABDTC$() | (Data creazione - sola lettura) |
| _ABDTU$() | (Data ultima modifica - sola lettura) |

Esempio:

Scrive un Codice Abbinato per un articolo, legge e scrive a video tutti i codici Abbinati e infine cancella tutti quelli di un altro articolo

```
AZZVARSYS 22
_ABART$(1)="ARTICOLO 1" : _ABCOD$(1)="CODICEABBINATO"
_ABPRO(1)=-1 : _ABCAT$(1)="" : _ABCATD$(1)=""
PUTABBIN
IF _ERRAB<>0 THEN VIMSG 5, _ERRAB$
GETABBIN "","",-1,""
IF _ERRAB<>0 THEN VIMSG 5, _ERRAB$
FOR I=1 TO _NAB
LPRINT "CODART       :" ;   : LPRINT _ABART$(I)
LPRINT "CODAB        :" ;   : LPRINT _ABCOD$(I)
LPRINT "CATEGORIA    :" ;   : LPRINT _ABCAT$(I)
LPRINT "DESCRIZ CATEG  :" ;   : LPRINT _ABCATD$(I)
LPRINT "PROG         :" ;   : LPRINT _ABPRO(I)
  LPRINT "DATA CREAZ    :" ;   : LPRINT _ABDTC$(I)
  LPRINT "DATA ULT MODIF :" ;   : LPRINT _ABDTU$(I)
NEXT
DELABBIN "ARTICOLO 2","",-1,"" : IF _ERRAB<>0 THEN VIMSG 5, _ERRAB$ : END
```

## _ACCFILEUTE

In Sprix/Collage determina la gestione degli accessi a file e directory tramite istruzioni sprix, da parte degli utenti definiti nel Servizio Utenti e Chiavi di accesso, in cui stabilire directory consentite e/o vietate. Se = 1avviene il controllo, se = 0 gli accessi sono liberi per tutti gli utenti come se nel Servizio non fosse definita alcuna directory (salvo permessi stabiliti a livello file system del Server). Il controllo avviene anche per le righe di stampa (di tipo 'F') dell'editor.

La gestione di file di tipo utente _ACCFILEUTE=1 valorizza _ERRFILE$ che contiene il relativo errore di accesso non consentito; l'errore su directory è valorizzato da LEGGIDIR. Per regolamentare l'accesso ai record del file utilizzare la semaforica.

ATTENZIONE: in tutte le operazioni che richiedono il nome del file, se questo non è completo di directory (percorso) viene aggiunta automaticamente laeventuale directory specificata come parametro alla partenza del gestionale (ad esempio nella dati); la lunghezza del nome del file compresa l'eventuale directory è max 200 caratteri (compreso l'eventuale '<'); alcuni sistemi operativi distinguono fra caratteri MAIUSCOLI e minuscoli.

# _AD

variabili struttura archivi Associazione Gest.-Docuv. (GET-PUT-DEL)

| _ADDVDID() | Numero documento; sola lettura |
|---|---|
| _ADTIPO() | Classi delle associazioni; sola lettura |
| **PIANO DEI CONTI** | |
| _ADPCCOD$() | Codice conto |
| _ADPCTIPC$() | Tipo conto C/F |
| _ADPCCAL$() | Codice alternativo |
| _ADPCRIC$() | Nome ricerca |
| _ADPCPRO$() | Provincia |
| _ADPCONA() | Zona |
| _ADPCCST() | Categoria statistica |
| **PRIMA NOTA** | |
| _ADPNSAZ() | Codice SottoAzienda |
| _ADPNDRE$() | Data di registrazione |
| _ADPNPRN() | Progressivo di registrazione |
| _ADPNCAU$() | Causale |
| _ADPNTDE$() | Descrizione |
| _ADPNRPR$() | Tipo Registro A/V/C |
| _ADPNSPR() | Serie protocollo |
| _ADPNNPR() | Numero protocollo |
| _ADPNNDO() | Numero documento |
| _ADPNDDO$() | Data documento |
| **Articoli** | |
| _ADARCOD$() | Codice articolo |
| _ADARDES$() | Descrizione articolo |
| _ADARALT$() | Codice alternativo |
| _ADARSTA$() | Categoria statistica |
| **DOCUMENTI DI MAGAZZINO** | |
| _ADMMSAZ() | Codice SottoAzienda |
| _ADMMSTPD() | Documento stampato |
| _ADMMSIG$() | Sigla documento |
| _ADMMSER() | Serie documento |
| _ADMMNUM() | Numero documento |
| _ADMMDAT$() | Data documento |
| ADMMANFIS() | Anno fiscale; solo doc. movimento |
| _ADMMCLI$() | Codice conto |
| _ADMMZON() | Zona conto; solo doc. movimento |
| _ADMMSTA() | Categoria statistica; solo doc. movimento |
| _ADMMNOT$() | Note |
| **BOLLE DI LAVORAZIONE** | |
| _ADOBSTPD() | Documento stampato |
| _ADOBNUM() | Numero bolla |
| _ADOBNSO() | Numero sottobolla |
| _ADOBDAT$() | Data bolla |
| _ADOBCTO$() | Codice conto |
| _ADOBDST$() | Nota |
| **NOTULE / PARCELLE** | |
| _ADNPSAZ() | Codice SottoAzienda |
| _ADNPSTPD() | Documento stampato |
| _ADNPSIG$() | Sigla documento |
| _ADNPSER()) | Serie documento |
| _ADNPNUM()) | Numero documento |
| _ADNPDAT$() | Data Documento |
| _ADNPCLI$() | Codice conto |
| _ADNPNOT$() | Note |
| **TIPO ALLEGATO** | |
| _ADARTPA$() | Tipo Allegato*a |
| _ADARPRGA() | Progressivo Immagine*a |

*Note:*

*a - modificabili solo con parametro [alleg] di PUTASSOCDV

Esempio: registrazione di associazione documento Matrice del Gestionale al documento Docuvision

```
AZZVARSYS 25
_ADMMSIG$(1)="MA":_ADMMSER(1)=1:_ADMMNUM(1)=290
_ADMMDAT$(1)="01012021":_ADMMCLI$(1)="201.00028"
PUTASSOCDV 1500,"MA;1;290;201.00028;20210101",1
IF _ERRAD<>0 THEN VIMSG 4, _ERRAD$
```

# _AE

variabili di struttura archivi di Anagrafiche Aree (GET-PUT)

| _AECOM$ | Codice Area |
|---|---|
| _AEDES$ | Descrizione |
| _AEPAD$ | Codice Area padre |
| _AESOS$ | Cod. Area Sost. |
| _AETPA$ | Tipo Area |
| _AETPZ$ | Zona |
| _AERES$ | Responsabile |
| _AENOT$ | Nota |
| _AEDTV$ | Data Validità |

| _AEDTC$ | Data creazione |
|---|---|
| _AEDTM$ | Data ult.agg |
| _AEANN$ | Annullato |

Esempio:
legge un'area e le sostituisce la nota

```
AZZVARSYS 24 : GETAREA "1A00" : IF _ERRAE$<>"" THEN VIMSG 4,_ERRAE$
VIMSG 1, "Nota dell'area = " + _AENOT$ : _AENOT$="Nuova nota"
PUTAREA : IF _ERRAE$<>"" THEN VIMSG 4,_ERRAE$ ELSE GOTO MSG
MSG: VIMSG 1, "Ho modificato con nuova Nota ="+_AENOT$
END
```

# _AGP

variabili di struttura archivi AGENDA DI LAVORO (GET-PUT-DEL)

| _AGPNELE | Elementi trovati con la GETAGP |
|---|---|
| _AGPDTLAV$() | Data lavori (campo chiave) |
| _AGPID() | Numero record (campo chiave, non visibile esternamente) |
| _AGPSTATO$() | Stato del lavoro A/F/P |
| _AGPPRCOD$() | Codice della prestazione |
| _AGPDATAPA$() | Riporta data Prestazione su notula/Parcella S/N |
| _AGPNOTAPA$() | Riporta Nota su No/PA N/V |
| _AGPSPESA() | Spesa anticipata |
| _AGPCODCLI() | Codice Cliente |
| _AGPNOTE$() | Nota |
| _AGPAZIE$() | Sigla azienda |
| _AGPPRZ() | Prezzo |
| _AGPQTA() | Quantità |
| _AGPTIPOR$() | Origine (Stato=A/F/P/N/S) |
| _AGPAZZERA$() | S/N |
| _AGPUTE() | Id Utente |

```
FOR K = 1 TO 100
 AZZVARSYS(34)
 GETAGP K,"","","","","","","",""
 LPRINT _ERRAGP$+" ---- "+STR$(K,0,0)+" --- "
 LPRINT _ERRAGP
  FOR I = 1 TO _AGPNELE
   LPRINT _AGPDTLAV$(I)
   LPRINT _AGPID(I)
   LPRINT _AGPSTATO$(I)
   LPRINT _AGPPRCOD$(I)
  NEXT
NEXT
```

# _AL

variabili di struttura archivi Codici Alias Articoli (GET-PUT-DEL)

| _ALPRO() | Numero progressivo; Specifico = 0,1,2,..; AUTO=-1 |
|---|---|
| _ALCOD$() | Codice Alias |
| _ALART$() | Codice articolo |
| _ALDES$() | Caratteristiche |
| _ALNO1$() | Nota 1 |
| _ALNO2$() | Nota 2 |
| _ALDTC$() | Data creazione - solo lettura |
| _ALDTU$() | Data aggiornamento - solo lettura |

Esempio: Scrive un Codice Alias per un articolo, lo legge e gli modifica le caratteristiche e cancella tutti gli Alias di un altro articolo

```
AZZVARSYS 21
_ALART$(1)="ARTICOLO 1" : _ALCOD$(1)="ALIAS"
_ALDES$(1)="Caratteristiche Alias" : _ALPRO(1)=-1
PUTALIAS : IF _ERRAL<>0 THEN VIMSG 4, _ERRAL$
GETALIAS "","",-1 : IF _ERRAL<>0 THEN VIMSG 4, _ERRAL$
FOR I=1 TO _NAL
  IF _ALCOD$(I)="ALIAS" THEN GOTO CAMBIA
NEXT
CAMBIA: _ALDES$(I)="Nuove Caratteristiche Alias"
PUTALIAS : IF _ERRAL<>0 THEN VIMSG 4, _ERRAL$
DELALIAS "ARTICOLO 2","",-1: IF _ERRAL<>0 THEN VIMSG 4, _ERRAL$ : END
```

# _AR

variabili di struttura archivi Anagrafica Articoli (GET-PUT-DEL)
In rosso evidenziati i campi minimi obbligatori

| ANAGRAFICA GENERALE | |
|---|---|
| _ARCOD$ | Cod.articolo; se EAN il "+" alla fine e' check |

| _ARNRA | Numero record articolo = 0 Nuovo articolo se <>0 indica la variazione dell'articolo Sola Lettura |
|---|---|
| _ARTIP$ | Tipo articolo |
| _ARSTR | Num.struttura |
| _ARDES$ | Descriz.articolo |
| _ARAGG$ | Descriz. aggiuntiva articolo |
| _ARALT$ | Cod.alt./ulteriore descrizione |
| _ARIVA$ | Aliquota iva |
| _ARUM1$ | Um.primaria |
| _ARUM2$ | Um.alt.;10-32 se T>9; modif.le se art.nuovo |
| _ARDEC | Numero decimali articolo |
| _ARTCO$ | Tipo confezione |
| _ARCON | Confezione articolo |
| _ARKOE | Coeff.conv/peso tara |
| _ARKES | Coeff.conv.esteso - solo lettura |
| _ARPTA | Peso tara |
| _ARVUC | Valuta costo ultimo anno corrente |
| _ARCUC | Costo ultimo anno corrente - solo lettura |
| _ARVST | Valuta costo standard |
| _ARSST | Costo standard |
| _ARFOR$(3) | Codice fornitore |
| _ARMFT | Quantita' minima fatturabile |
| _ARMIN | Importo minimo fatturabile |
| _ARMAX | Importo massimo fatturabile |
| _ARCPS | Controllo peso |
| _ARQPR | Pr/Pa quantita' presa |
| _ARQPA | Pr/Pa quantita' pagata |
| _ARIMB$ | Imballo |
| _ARSCO | Tabella Sconti |
| _ARSCQ | Tab. Sconti q.ta' |
| _ARTPR | Tabella Provvigioni |
| _ARLIS | Categoria Formazione prezzi |
| _ARSTA$ | Cat. statistica; alfanumerico |
| _ARSTN | Cat. statistica; numerico |
| _ARDTC$ | Data creazione |
| _ARDTA$ | Data ult. aggiorn.to anagrafica -la conferma- |
| _ARDTAS$ | Data ultima modifica; solo tecnologia BP data di sistema e non data terminale |
| _ARRIC$ | Contropartita Ricavo |
| _ARCOS$ | Contropartita Costo |
| _ARRSO$ | Controp. Ricavo Sospeso, v. parcellazione |
| _ARANC$ | Analitico di Costo |
| _ARANR$ | Analitico di Ricavo |
| _ARPRE$ | Art. precancellato S/N |
| _ARANN$ | Art. annullato S/N |
| _ARPRU$ | Art. usato in PR - S/N - per GET/PUT/DEL |
| _ARSOS$ | Codice Articolo sostitutivo |
| _ARDBP$ | Art. con dbp S/N |
| _ARDBV$ | Art. con dbv S/N |
| _ARDBA$ | Art. con dba S/N - sola lettura |
| _ARCAR$ | Art. con cartella S/N |
| _ARTMO$ | Tipo modificatore articolo |
| _ARVMO | Valore modificatore |
| _ARSMO$ | Somma prz.base modificatore |
| _ARAMO$ | Tipo arrotondamento |
| _ARRRO | Valore arrotondamento |
| _ARRST | Riferimento struttura |
| _ARTDE$ | Tipo descrizione |
| _ARTPM$ | Tipo Valore: N=al Netto, F=Fisso |
| **COSTI ANNI PRECEDENTI** | |
| _ARVUP | Valuta costo ultimo anni precedenti |
| _ARCUP | Costo ultimo anni precedenti |
| _ARVUL | Valuta costo ultimo ultimo anno |
| _ARCUL | Costo ultimo ultimo anno |
| **FORNITORI** | |
| _ARGGR(3) | Giorni di riordino |
| _ARLOT(3) | Lotto di riordino |
| _ARVAL(3) | Valuta prezzo base |
| _ARFPR(3) | Prezzo base |
| _ARCOF$(3) | Cod. art. del fornitore |
| _ARQTA(3,3) | Fino a Quantita' |
| _ARSCQ$(3,4) | Sconto quantita' |
| _ARSCQN$(3,4) | Tab. Sconti q.ta' -cascata- |
| **SCONTI SU LISTINO** | |
| _ARSCO$(9) | Tab. Sconti; "*****" = non visibile |
| _ARSCON$(9) | Tab. Sconti - cascata - |
| **PROVVIGIONI** | |
| _ARPRO(9) | Provv.; solo GET in multiagente |
| _ARPROFORM$(9) | Provv. |
| **FORMAZIONE PREZZI LISTINO** | |
| _ARPER(9) | % Formazione prezzi su listino |
| **CAMPIONARIO** | |
| _ARCMA$(200) | Codice componente |
| _ARCMU$(200) | Unita' di misura |
| _ARCMQ$(200) | Quantita' |
| _ARCMT(200,16) | Quantita' Tg 1-16 |
| _ARCMTXL(200,16) | Quantita' Tg 17-32 |
| _ARCMP$(200) | Tipo condizione |
| _ARCMR$(200) | Condizione Art/Sml |
| _ARCMC$(200) | Condizione Cli |
| **TESTO** | |
| _ARTXV$(200) | Testo Variabile |
| _ARTXT$(200) | Testo Fisso |
| **NOTE** | |
| _ARNDS$(2) | Note |
| _ARNDE$(2) | Destinazione Nota |
| _ARNDT$(2) | Data scadenza Nota |
| **LINGUE** | |
| _ARDSL$(9) | Descriz. in Lingua |
| **LISTINI** | |
| _ARPRZ(9) | Prezzo |
| **EXTRA** | |
| _ARVEX | Cod. valuta Extra |
| _AREXG | Extra fisso |

| _AREXA | Extra a |
|---|---|
| _AREXB | Extra b |
| _AREXC | Extra c |
| **PRODUZIONE** | |
| _ARCTG$ | Art. consumo x taglia |
| _ARFRM$ | Art. di tipo forma |
| **GRUPPO MERCEOLOGICO** | |
| _ARGRPMER$ | Cod. Gruppo Merceologico |
| **NATURA** | |
| _ARNATURA$ | Codice Natura |
| **LOTTI** | |
| _ARRDT$ | Data inizio gestione rintracciabilita' |
| _ARRTL$ | Tipo lotto; variabile se non movim.to |
| _ARRRI$ | Richiesta lotto |
| _ARRSA$ | Selez.autom.variabile se non movim.to |
| **ENASARCO** | |
| _ARENAS$ | Gestione Enasarco La gestione Ritenuta d'Acconto e' riportata da _ARPRIT$ indicata nell'elenco di _AR relative alla Parcellazione-Prestazioni Clienti |
| **RATEI / RISCONTI** | |
| _ARPRR | Periodo in mesi del rateo Risconto |
| _ARGRR$ | Gestione rateo risconti -S/N- |
| **AGROALIMENTARE** | |
| _ARAGR$ | Agroalimentare; valori "A","D","" |
| **DATI MAGAZZINO** | |
| _ARUBI$ | Ubicazione |
| _ARSUB$ | Sotto ubicazione |
| _ARCAD$ | Ult.data di carico |
| _ARSCA$ | Ult.data di scarico |
| _ARORD$ | Ult.data ordini/imp |
| _ARDTM$ | Ult.data movimenti |
| _ARSMI | Scorta minima |
| _ARSMA | Scorta massima |
| **PROGRESSIVI ANNO/MAGAZZINO** | |
| _ARINQ | Qta inventario |
| _ARINT(16) | Qta inventario Tg 1-16 |
| _ARINTXL(16) | Qta inventario Tg 17-32 |
| _ARINV | Valore inventario |
| _ARTIN | Tara inventario |
| _ARCIN | Colli inventario |
| _ARCAR | Qta carico |
| _ARCAT(16) | Qta carico Tg 1-16 |
| _ARCATXL(16) | Qta carico Tg 17-32 |
| _ARCAV | Valore carico |
| _ARTCA | Tara carico |
| _ARCCA | Colli carico |
| _ARSCA | Qta scarico |
| _ARSCT(16) | Qta scarico Tg 1-16 |
| _ARSCTXL(16) | Qta scarico Tg 17-32 |
| _ARSCV | Valore scarico |
| _ARTSC | Tara scarico |
| _ARCSC | Colli scarico |
| **ORDINI** | |
| _ARORF | Ord.fornitori |
| _ARORT(16) | Ord.fornitori Tg 1-16 |
| _ARORTXL(16) | Ord.fornitori Tg 17-32 |
| _ARCLI | Clienti impegni |
| _ARCTI(16) | Clienti impegni Tg 1-16 |
| _ARCTIXL(16) | Clienti impegni Tg 17-32 |
| _ARCLS | Clienti sospesi |
| _ARCTS(16) | Clienti sospesi Tg 1-16 |
| _ARCTSXL(16) | Clienti sospesi Tg 17-32 |
| _ARCLA | Cl.autom/da produrre |
| _ARCTA(16) | Cl.autom/da prod.Tg 1-16 |
| _ARCTAXL(16) | Cl.autom/da prod.Tg 17-32 |
| _ARPRD | In produzione |
| _ARCTP(16) | In produzione Tg 1-16 |
| _ARCTPXL(16) | In produzione Tg 17-32 |
| **INTRACEE** | |
| _ARINO$ | Nomenclatura combinata |
| _ARIMA$ | Massa netta |
| _ARIKM | Coeff massa netta |
| _ARIUS$ | Unita' supplementare |
| _ARIDU$ | Descr. unita' suppl. |
| _ARIKU | Coeff. unita' suppl. |
| _ARIPR$ | Provincia origine |
| _ARIPA$ | Paese origine |
| _ARICS | Codice servizio |
| _ARIVS | Valore statistico unitario |
| **STATO ANAGRAFICA** | |
| _ARMRA$(12) | Stato anagr. attiva; v. mirror |
| _ARMRV$(12) | Stato anagr. variata; v. mirror |
| **IMMAGINI** | |
| _ARIMM$ | Immagine articolo*a |
| _ARCAT$ | Immagine catalogo*a |
| _ARICO$ | Immagine Icona*a |
| **VIDEATE ARTICOLO** | |
| _ARVU1$(26) | 26 vid. anagrafica non riportabile |
| _ARVU2$(26) | 26 vid. anagrafica riportabile |
| _ARWWW$(26,24) | 26 videate aggiuntive con 24 campi GET converte da num->str$, la PUT viceversa [SHF8] Videate |
| **PARCELLA PRESTAZIONI CLIENTI** | |
| _ARPPER$ | periodicita' |
| _ARPMES | periodicita' mese |
| _ARPRIT$ | ritenuta d'acconto |
| _ARPCAS$ | cassa previdenza |
| _ARGSI$ | Gestione separata INPS$ |

*Note:*

*a -Immagini articolo: le 3 variabili sono nel percorso dati aziendali, in una determinata cartella (es...\azi\imgart\42000000000000_c.bmp); leggibili con GETAR; in scrittura con PUTAR, l'azzeramento della variabile

cancella l'immagine associata, la non modifica della variabile non implica variazioni, la modifica dell'eventuale valore della variabile comporta la ricerca del file specificato (se trovato viene convertito nel nome atteso dal Gestionale, se non trovato avvisa che l'immagine non viene trovata e blocca la PUTAR).

Esempio:

Scrive articolo, rilegge, cambia descrizione e precancella

```
ART$="MELE" : MAG=1 : AZZVARSYS 3
_ARCOD$=ART$ : _ARTIP$="A" : _ARDES$="GOLDEN" : _ARIVA$="20"
_ARUM1$="kg" : _ARDEC=2 : _ARVST=2 : _ARVUL=2 : _ARVUP=2
PUTAR 1: IF _ERRAR<>0 THEN VIMSG 4, _ERRAR$
AZZVARSYS 3: GETAR ART$,MAG : IF _ERRAR<>0 THEN VIMSG 4, _ERRAR$
  _ARDES$="STARK": _ARANN$="S" : _ARSOS$="MELE DI SCARTO"
        PUTAR MAG: IF _ERRAR<>0 THEN VIMSG 1,_ERRAR$
        END
```

# _ARGCMD

## _ARGCMD$

solo in Sprix contiene l'argomento proveniente dal parametro -vdel <comandospx$>.

Es. lancio dello Sprix (spx101.es): VIMSG 1, _ARGCMD$:END

Lancio con Sprix:

```
ARG$=CHR$(34)+"Pass 123"+CHR$(34) 'se senza blank ARG$="Pass123"
CMD$= sxdft + " -t0 -o0 -x01032009 -aAZI -pspx101 -v"+ARG$
SHELLSPRIX CMD$ : END
```

Lancio Collage:

```
ON_612_ALL:
ARG$="Pass123"
CMD$=" -t0 -o0 -x"+STRDATA$(sxdat,3,"")+" -a"+azsig+" -v"+ARG$+" -p"
SHELLSPRIX CMD$+"#101" : END
```

Lo schedulator del sistema operativo (Operazioni Pianificate, Cron, ecc..) consente esecuzioni temporizzate (es. dal prompt dei comandi), di appositi eseguibili Passepartout (come ad esempio l'esecutore a riga di comando) in grado di lanciare SPRIX analogamente a quanto avviene con la SHELLSPRIX. Verificare sempre i riferimenti a percorsi esistenti.

# _BUFCOM

## _BUFCOM$

Buffer di memorizzazione dei byte letti da GETCOM

Se mancano caratteri da leggere, GETCOM attende il <timeout> di OPENCOM. La GETCOM valorizza _ERRCOM$.

# _CDG

variabili di struttura archivi di Contabilità Analitica (GET-PUT-DEL)

| | |
|---|---|
| _CDGTIPREG$(N) | Tipo Registrazione:<br>A=MATRICI<br>P=PREVENTIVI<br>E=EXTRA<br>R=PRIMANOTA<br>M = MOV. MAGAZZINO<br>F=ORD.FORNITORE<br>C=ORD.CLIENTI |
| _CDGSN$(N) | Considera movimento su CDG S/N |
| _CDGDDTCOMP$(N) | Data Competenza |
| _CDGSCODANL$(N) | Codice Analitico |
| _CDGSCOMM$(I) | Codice Commessa |
| _CDGSAREA$(I) | Codice Area |
| _CDGCTOCG$(I) | Codice Contropartita righe dei movimenti |
| _CDGCODARMO$(N) | Cod. Articolo Movimento |
| _CDGSIGDOC$(N) | Sigla Documento di magazzino |
| _CDGISERDOC(N) | Serie Documento di magazzino |
| _CDGLNUMDOC(N) | Numero Documento di magazzino |
| _CDGDDTDOC$(N) | Data Documento di magazzino |
| _CDGCTOT$(N) | Conto Cliente/Fornitore Documento |
| _CDGNTEST(N) | Num.testata documento di origine di riga; se diverso da uno richiede _CDGPROGRIG(N) 'equivalente a _MMNCRERIGA(N) |
| _CDGCAUPRN$(N) | Causale movimento di primanota |
| _CDGCPROT$(N) | Protocollo -Registro- |
| _CDGIPROTS(N) | Serie protocollo |
| _CDGLPROTN(N) | Numero protocollo |
| _CDGDDTREG$(N) | Data registrazione di prima nota |
| _CDGNDOCPN(N) | Numero di documento di prima nota |
| _CDGNOTA$(N) | Nota riga controllo di gestione |
| _CDGFVALG(N) | Importo valuta di gestione |
| _CDGIDEC(N) | Numero di decimali gestiti in variabile quantità |
| _CDGFQTA(N) | Quantità |

| | |
|---|---|
| _CDGSTRIPR$(N) | Stato riga preventivo:<br>T trasformata<br>' '-spazio- Sospesa |
| _CDGCATSTA$(N) | Tipo movimento CDG presente sul movimento -<br>' ' default come tabella automatismi causali contabili/docum.magazzino<br>- altrimenti valore inserito dall'utente |
| _CDGIVAL(N) | Valuta Importo di sola lettura |
| _CDGFVAL(N) | Importo in valuta di sola lettura |
| _CDGPROGANA(N) | numero progressivo -numero riga- relativo all'analitico |
| _CDGPROGRIG(N) | numero riga documento<br>- in caso di prima nota è il numero di riga visibile a video<br>- per i movimenti di magazzino è il valore rilasciato dalla _MMNCRERIGA o dalla mmrig() in caso di trasformazione dei documenti, altresi' è il numero di riga del documento |
| _CDGPROGEXT(N) | progressivo per extra, da gestire solo per movimenti extra |
| _CDGRDOCSOS(N) | numero riga di documenti sospesi è il valore rilasciato da_MMNCRERIGA o dalla mmrig() |
| _CDGRNPRPN(N) | numero primo record assegnato -da 1 a 16777214- |
| _CDGTIPEXT$(N) | tipo di Extra P/M/G |
| _CDGCESCOD$(N) | Codice Cespite |
| _CDGCESANNO(N) | Anno Cespite |
| _CDGCESCONT$(N) | Conto di contabilità relativo al Cespite |

Esempio Sprix: letture e cancellazioni di movimenti di analitica

```
GETCDG "R","PNDRE=20060906;PNNDO=19;",2006,2,0
DELCDG "R","PNDRE=20060906;PNNDO=19;",2006,2,1
GETCDG "M","MMSIG=SL;MMNUM=7;MMSER=1;MMCLI=501.1;NTMM=1;",0,0,0
DELCDG "M","MMSIG=FT;MMNUM=7;MMSER=1;MMCLI=501.1;NTMM=1;",0,1,0
GETCDG "C","MMSIG=OC;MMNUM=1;MMSER=1",0,0,0
GETCDG "F","MMSIG=OF;MMNUM=1;MMSER=1",0,1,0
GETCDG "E","PRGEXTRA=;DTCOMP=20081231;AREA=A;COMM=C;CODANL=C1;TIPMOV=M;",...
DELCDG "E","PRGEXTRA=;DTCOMP=20080909;AREA=A;COMM=C;CODANL=C1;TIPMOV=G;",...
```

# _CFR

variabili di struttura archivi dei Referenti (GET-PUT-DEL)

| | |
|---|---|
| _CFRCOD$(1-9) | Codice del Cliente/Fornitore |
| _CFRPRG(1-9) | Codice Referente |
| _CFRPOS(1-9) | Cod.Posizione Referente in tabella posizioni |
| _CFRNOME$(1-9) | Nome Referente |
| _CFRNOTA$(1-9) | Nota Referente |
| _CFRTEL$(1-9) | Telefono Referente |
| _CFRCELL$(1-9) | Cellulare Referente |
| _CFRFAX$(1-9) | Fax Referente |
| _CFRMAIL$(1-9) | Indirizzo e-mail Referente |
| _CFRWEB$(1-9) | Indirizzo Internet Referente |
| _CFRTP$(1-9) | Tipo Referente R= Cli/For, O=Contatti |

Esempio Sprix REFERENTI

Nel Cliente 501.00001 inserisce un referente di segeteria in posizione 2

```
_CFRCOD$(1)="501.00001" : _CFRPRG(1)=2 : _CFRPOS(1)=2
_CFRNOME$(1)="Segretario Rossi" : _CFRNOTA$(1)="nota..."
PUTCFR : IF _CFRERR<>0 THEN VIMSG 1, _CFRERR$
```

# _CFRERR

La _CFRERR e la _CFRERR$ riportano il codice e la descrizione dell'esito dell'operazione (GET/PUT/DEL) sulle variabili _CFR?. Ai messaggi è anteposto"Rigo: nn" dove nn = elemento di array.

| | |
|---|---|
| 0 | esito positivo dell'operazione (nessun messaggio, il campo è vuoto) |
| 1 | Referente assente (in GET) |
| 1 | Cod cli/for [cod.] intestatario non trovato in anagrafica<br>cioè non codificato, in PUT |
| 1 | Cod cli/for intestatario non specificato<br>non valorizzato su array 1, in PUT |
| 2 | Cod cli/for intestatario diverso da precedente<br>(array con _CFRCOD$ diversi, in PUT) |
| 3 | Codice referente [codice] errato, (ammessi da 1 a 9) (in PUT) |
| 4 | Codice referente [codice] già presente su rigo [rigo] (in PUT) |
| 5 | Dati referente non valorizzati (in PUT) |
| 6 | Posizione referente già assegnata ad altro referente (in PUT) |
| 7 | Tipo referente errato (accettati 'R' per Cli/For, 'O' per Contatti (in PUT) |
| 10 | Referenti assenti (in DEL con parametro <tipo> uguale a zero) |
| 11 | Codice referente non esistente (in DEL con param. <tipo> <> da zero) |
| 12 | Codice referente errato (in DEL con parametro <tipo> diverso da zero) |

# _CO

Variabili di struttura archivi di Anagrafiche Commesse (GET-PUT)

| ANAGRAFICA GENERALE | |
|---|---|
| _COCOM$ | Codice |
| _CODES$ | Descrizione |
| _COPAD$ | Codice Padre |
| _CODEA$ | Descrizione Aggiuntiva |
| _CONOT$ | Note |
| _COCLI$ | Cliente |
| _COMER$ | Destinatario Merce |
| _CORES$ | Responsabile |
| _COPAG | Cod. pagamento |

| | |
|---|---|
| _COSTA$ | Stato Commessa |
| _COANN$ | Commessa Sostitutiva |
| _CODTP$ | Data Preventivo |
| _CODTA$ | Data Apertura |
| _CODTO$ | Data Ordine |
| _CODTS$ | Data Spedizione |
| _CODTG$ | Data Chiusa in Garanzia |
| _CODTD$ | Data Chiusura |
| _CODTC$ | Data Creazionione |
| _CODTU$ | Data Ultimo Aggiornamento |
| **PAGAMENTI** | |
| _COTPS$(250) | Tipo Pagamento |
| _COTPR$(250) | % o valore |
| _CORAP(250) | % |
| _CORAV(250) | Valore |
| _COSCA$(250) | Scadenza |
| _COTPP$(250) | Stato Pagamento |
| _CODEP$(250) | Descrizione |
| **CONDIZIONI COMMERCIALI** | |
| _COAGE$ | Agente |
| _COPPR | % Provvigion |
| _COTPCOM$ | Tipo |
| _COZONA$ | Zona |
| **CONTRATTO** | |
| _COCDT$ | Data contratto |
| _CO1DT$ | Data 1^collaudo |
| _CO2DT$ | Data 2^collaudo |
| _COLDT$ | Data consegna lavori |
| _COANT | % Anticipo |
| _CORIC | % Ricarico Prezzo |
| _COADT$(3) | Data Anticipo |
| _COAIM(3) | Anticipo |
| _COAPE(3) | Penale |
| _COADE$(3) | Descrione |
| **ALLEGATI** | |
| _COALL$(4) | Allegati |

Esempio:
legge una commessa e le sostituisce la commessa sostitutiva

```
AZZVARSYS 23 : GETCOMM "COMM1A3" : IF _ERRCO$<>"" THEN VIMSG 4,_ERRCO$
VIMSG 1, "Commessa Sostitutiva = " +_COANN$: _COANN$="COMM1A1"
PUTCOMM : IF _ERRCO$<>"" THEN VIMSG 4,_ERRCO$ ELSE GOTO MSG
MSG: VIMSG 1, "Ho modificato con nuova Commessa Sostitutiva = " +_COANN$
END
```

# _DA

Variabili struttura archivi Anagrafica Indirizzi spedizione (GET-PUT)

| | |
|---|---|
| _DACTO$ | Codice conto Cli/For |
| _DACIN$ | Cod. Indirizzo o AUTO |
| _DADES$ | Descrizione |
| _DAIND$ | Indirizzo |
| _DACAP$ | C.A.P. |
| _DALOC$ | Località |
| _DAPRO$ | Provincia |
| _DANAZ$ | Nazionalità |
| _DAPAE$ | Paese ISO2 |
| _DATEL$ | Telefono |
| _DAFAX$ | Fax |
| _DAMAI$ | Indirizzo e-Mail |
| _DAINT$ | Indirizzo Internet |
| _DACHI$ | Chius. sett. o stringa libera |
| _DACON$ | Riferimento |
| _DACEL$ | Cellulare Riferimento |
| _DANOT$ | Note |
| _DAALL$ | Nome file allegato |
| _DADVA$ | Data validità |
| _DADCR$ | Data creazione |
| _DADAG$ | Data aggiornamento |
| _DACTS$ | Cod.conto sostitutivo |
| _DACIS$ | Cod.indirizzo sostitutivo |
| _DAANN$ | Annullato |
| _DARAG$ | Raggruppa documenti S/N |
| _DACIG$ | Codice CIG |
| _DACUP$ | Codice CUP |

Esempio:
registra un Anagrafica Indirizzo di Spedizione e poi la aggiorna

```
_DACIN$="AUTO" : _DACTO$="201.00002"
_DADES$="descrizione indirizzo":_DANAZ$="I":_DAPAE$="IT"
PUTIND : IF _ERRDA<>0 THEN VIMSG 1, _ERRDA$
CODIND$=_DACIN$
IF _ERRDA=0 THEN VIMSG 1, "dopo la scrittura|aggiorno la descrizione"
AZZVARSYS 12: GETIND CODIND$,"": IF _ERRDA<>0 THEN VIMSG 1, _ERRDA$
_DADES$="NUOVA Descrizione"
PUTIND : IF _ERRDA<>0 THEN VIMSG 1, _ERRDA$
END
```

# _DC

Variabili struttura archivi Anagrafica Contatti

| | |
|---|---|
| _DCCTO$ | Codice contatto |
| _DCINT$ | Indirizzo Internet |
| _DCDES$ | Descrizione |
| _DCNOT$ | Note |
| _DCCOG$ | Cognome |
| _DCALL$ | Nome file allegato |
| _DCIND$ | Indirizzo |
| _DCDVA$ | Data validità |
| _DCCAP$ | C.A.P. |
| _DCDCR$ | Data creazione |
| _DCLOC$ | Località |
| _DCDAG$ | Data aggior.anagr. |
| _DCPRO$ | Provincia |
| _DCPIV$ | Partita iva |
| _DCNAZ$ | Nazionalità |
| _DCCFI$ | Codice fiscale |
| _DCPAE$ | Paese Iso |
| _DCPFI$ | Persona fisica S/N |
| _DCTEL$ | Telefono |
| _DCRIC$ | Nome di ricerca |
| _DCFAX$ | Fax |
| _DCPRI$ | Privato S/N |
| _DCMAI$ | Indirizzo e-Mail |
| _DCCODPC$ | codice piano dei conti se = 0 ritorna nelle variabili _DCPCCLI$ _DCPCFOR$ i codici a cui l'anagrafia fa riferimento se = 1 anagrafica non collegata altrimenti riporta il codice del piano dei conti a cui è collegata |
| _DCPCCLI$ | se _DCCODPC$ = 0 riporta il codice conto clientesolo OUTPUT |
| _DCPCFOR$ | se _DCCODPC$ = 0 riporta il codice conto fornitore dolo OUTPUT |

Esempio:

registra un Anagrafica Contatto e lo duplica modificandone il nome di ricerca e infine cancella il vecchio contatto

```
_DCCTO$="AUTO" : _DCNAZ$="IT" : _DCDES$="Descrizione Contatto"
_DCRIC$="Nome di Ricerca"
PUTCON : IF _ERRDC<>0 THEN VIMSG 4, _ERRDC$
CODCONT$=_DCCTO$ : AZZVARSYS 13
VIMSG 1, "duplica il contatto e lo modifica|poi cancella l'originario"
GETCON CODCONT$ : IF _ERRDC<>0 THEN VIMSG 4, _ERRDC$
_DCCTO$="AUTO" : _DCNAZ$="IT" : _DCDES$="Contatto Duplicato"
_DCRIC$="NUOVO NOME DI RICERCA"
PUTCON : IF _ERRDC<>0 THEN VIMSG 4, _ERRDC$
DELCON CODCONT$ : IF _ERRDC<>0 THEN VIMSG 4, _ERRDC$
END
```

# _DV

variabili di struttura archivi Documenti di Docuvision (GET-PUT-DEL)

| | DATI DOCUMENTO |
|---|---|
| _DVDID | Num. documento; 0=AUTO ***a** |
| _DVDCLASS | Classe documento |
| _DVDTIT$ | Titolo documento |
| _DVDDATA$ | Data documento |
| _DVDPROTO$ | Numero protocollo |
| _DVDDATPRO$ | Data protocollo |
| _DVDCNT$ | Cliente/fornitore |
| _DVDCOGN$ | Ragione sociale o cognome |
| _DVDNOME$ | Nome |
| _DVDPIVA$ | Partita iva |
| _DVDCFI$ | Codice fiscale |
| _DVDNOTA$ | Nota |
| _DVDAZIE$ | Sigla azienda; solo GET |
| _DVDALLCL$ | Classi a cui il doc. è allegato; solo GET |
| _DVDUFFPA$ | Codice ufficio PA |
| _DVDSTDOC$ | Stato trasmissione documento<br>(accetta: I=Inviato,A= Archiviato,R= Rifiutato, C=Conservato, =blank) |
| _DVDIDPACCH | Id invio pacchetto |
| _DVDTPCONTR$ | Tipo contratto |
| _DVDDTCOMIN$ | Data competenza contratto |
| _DVDTPCLASS$ | Tipo documento della classe |
| DATI REVISIONE | |
| _DVVNUM() | Numero revisione ***b** |
| _DVVESTFIL$() | Estensione file |
| _DVVNPAG() | Numero di pagina del documento *d |
| _DVVDATA$() | Data revisione ***f** |
| _DVVAUTOR$() | Autore ***f** |
| _DVVNOTA1$() | Campo nota 1 ***e** |
| _DVVNOTA2$() | Campo nota 2 ***e** |
| _DVVVOL() | Numero volume; solo GET |
| _DVVFILEGET$() | Nome file; solo GET |
| _DVVFILEPUT$() | Nome file; solo PUT ***c** |
| _DVVIDAUT() | ID autore revisione ID=utente passepartout |
| _DVVFIRMA$() | Firma revisione |
| _DVVMARCA$() | Marca revisione |
| _DVVDTRAS$() | Data trasmissione |
| _DVVTPRIC$() | Tipo ricevuta |
| _DVVOTRAS$() | Ora data trasmissione nella forma hhmmss accetta AUTO |
| _DVVORA$() | Ora data revisione nella forma hhmmss accetta AUTO |
| DATI CAMPI AGGIUNTIVI | |

| | |
|---|---|
| _DVAVAL$() | Valori campi |
| _DVANOMECOL$() | Nome colonna; solo GET |
| _DVANCAMPI() | Numero di campi |
| _DVVIDEST | Id documento docuvision AutoOn ***g** |

*Note:*

***a** - dopo la registrazione AUTO contiene il numero attribuito (da leggere); specificare la _DVDID equivale ad aggiornare il documento (se _DVDID non esiste rilascia errore) altrimenti, se non è specificata equivale a inserire un documento nuovo;

***b** - con gestione delle pagine attiva è inutile impostarla (l'elaborazione è sempre sull'ultima revisione); se attiva la gestione delle revisioni,specificandola equivale ad aggiornare una revisione del documento;

***c** - non èallegabile un nome file nuovo a una revisione già esistente (consentito invece inserendo una nuova revisione); se non è attiva la gestione delle revisioni, allegare un file senza specificare _DVVNUM() equivale a sostituire l'allegato all'ultima revisione presente nel database.

***d** - se è attiva la gestione delle pagine, l'indicazione in scrittura sostituisce l'allegato (alla pagina indicata e già allegata al documento) con il file indicato in _DVVFILEPUT$(); specificare _DVVNPAG=0 equivale ad aggiungere una pagina, altrimenti non scrive il documento e genera errore);

***e** - richiesta in scrittura, ma non gestite con la gestione pagine attiva;

***f** - solo GET (e per l'autore se gestite le chiavi accesso); in scrittura sono ignorate e impostate automaticamente (aggiornate sul database in caso di sostituzione di tutte le pagine allegate a documento); con la modifica parziale delle pagine allegate, viene verificata la compatibilità con quelle già esistenti (la tipologia di file), altrimenti rilascia errore e non avviene la scrittura.

***g** - solo per documenti residenti su database di installazione 'AutoOn'.

Per GETDVDOC e PUTDVDOC (PUT può anche inserire documenti esterni e vale solo in installazione commercialista). La variabile contiene l'Id documento del documento docuvision, per trasferire la prima nota con documenti allegati in docuvision, senza che siano trasmessi gli allegati, ma solo un collegamento ad essi e che quindi sono richiamabili pur risiedendo solo sull'installazione AutoOn. È un collegamento tra un elemento di docuvision del commercialista e il documento reale dell'azienda, per i soli documenti indicati come da inviare. Esempio: l'azienda AutoOn spedisce al commercialista una registrazione di prima nota a cui e allegato il documento con Id 25 sul database locale alla AutoOn. Quando il commercialista carica la registrazione, crea un documento fittizio con un suo Id univoco sul database del commercialista (ad esempio 47) il cui record contiene il valore 25 nel campo leggibile attraverso la _DVVIDEST.

Attenzione, le modifiche devono rispettare i vincoli di gestione del documento e delle revisioni per come impostati nei Parametri Generali Docuvision:

- non è inseribile una revisione/pagina nuova o aggiornabile una revisione/pagina del documento, senza contemporaneamente scrivere anche il documento impostandone le variabili, anche se i dati del documento sono letti e non modificati;
- la revisione del documento è fino alla prima in cui non sono specificate le variabili _DVVNUM (aggiornamento di revisione esistente) e _DVVFILEPUT$ (con Id revisione a zero, ossia nuova revisione); in ingresso non è letta la _DVDNUMVER, ma viene calcolata;
- nell'aggiornamento di un documento indicato come non modificabile (dai Parametri Generali di Docuvision) i valori impostati nelle variabili devono essere identici a quelli presenti sul database, compresa la sezione dei campi aggiuntivi (se previsti);
- la scrittura del documento nuovo senza specificare gli eventuali campi aggiuntivi collegati, genera il documento con campi aventi valori di default se alcuni di questi sono previsti con valori di default (anche in modo non obbligatorio) dalla classe del documento, altrimenti non li genera (se nessuno ha valori di default); se il documento non ènuovo e qualche campo è obbligatorio, la loro non specificazione genera errore, altrimenti, se nessuno è obbligatorio, vengono cancellati

```
Esempio: crea nuovo documento sovraziendale (ricercabile in archivio senza azienda) con 2
nuove revisioni; legge e modifica le note delle revisioni (nb: Parametri Docuvision = gestione
revisioni attiva e modifica di documento fino ad archiviazione). La lettura delle pagine e la
cancellazione non sono attive (vanno impostati i parametri).
_DVDID=0 : _DVVNUM(1)=0 : _DVVNUM(2)=0
_DVDCLASS=1400 : _DVDTIT$="Titolo classe" : _DVDDATA$=sxdat
_DVVNOTA1$(1)="nota rev.1":_DVVNOTA2$(1)="ult.nota rev.1"
_DVVFILEPUT$(1)="c:\file1.txt"
_DVVNOTA1$(2)="nota rev.2":_DVVNOTA2$(2)="ult.nota rev.2"
_DVVFILEPUT$(2)="c:\file2.txt"
PUTDVDOC : IF _ERRDV<>0 THEN VIMSG 4, _ERRDV$
ID=_DVDID
VIMSG 1,"ha creato un Documento sovraziendale n. "+STR$(ID,0,0)

AZZVARSYS 20
GETDVDOC ID,-1: IF _ERRDV<>0 THEN VIMSG 4, _ERRDV$
```

```
'FOR I=1 TO _DVDNUMVER 'numero globale revisioni
'VIMSG 1,"rev.glob.: "+STR$(_DVVNUM(I),0,0)+" n.rev.: "+STR$(_DVDNUMVER,0,0)
'VIMSG 1, STR$(_DVDPAGXVER(I),0,0)'gestione pagine
'VIMSG 1, "DVVNPAG "+STR$(_DVVNPAG(I),0,0)      'gestione pagine
'VIMSG 1, "nome file pagina "+_DVVFILEGET$(I)    'gestione pagine
'NEXT

_DVVNOTA1$(2)="NOTA REV.2": _DVVNOTA2$(2)="ULT NOTA REV.2"
 PUTDVDOC : IF _ERRDV<>0 THEN VIMSG 4, _ERRDV$
'DELDVDOC 3,0: IF _ERRDV<>0 THEN VIMSG 4, _ERRDV$ 'cancella intero doc.
END
```

# _ERRAB

### _ERRAB

### _ERRAB$

In Sprix azzera le variabili di struttura archivi _AB? dei Codici Abbinati.

_ERRAB e _ERRAB$ riportano in Sprix codice e descrizione del esito dell'operazione (GET/PUT/DEL) sulle variabili_AB? :

| | |
|---|---|
| 0 | esito positivo dell'operazione (nessun messaggio, il campo e vuoto); |
| se <>0 | errore descritto in _ERRAB$; |

# _ERRAD

### _ERRAD

### _ERRAD$

In Sprix azzera le variabili di struttura archivi _AD? delle Associazioni di Documenti Gestionale-Docuvision.

La _ERRAD e la _ERRAD$ riportano in Sprix il codice e descrizione dell'esito dell'operazione (GET/PUT/DEL) sulle _AD?:

| | |
|---|---|
| 0 | Esito positivo dell'operazione, nessun messaggio, il campo è vuoto; |
| 1 | Errore generico con relativa descrizione; |
| 2 | Funzione non eseguibile; |
| 3 | Associazioni non trovate; |
| 4 | Entità in uso da altri terminali; |
| 7 | Parametri errati; |

# _ERRADCOM

### _ERRADCOM

### _ERRADCOM$

La _ERRADCOM e la _ERRADCOM$ riportano in Sprix il codice e descrizione dell'esito dell’istruzione ADCOMMERCIALE:

| | |
|---|---|
| 0 | Esito positivo dell'operazione, nessun messaggio, il campo è vuoto; |
| 1 | Parametro 'Modalità aggiornamento' errato o assente. Valori accettati 'C'- completa o 'V'- solo variati |
| 2 | Parametro 'Movimenti di magazzino' errato o assente. Valori accettati 'S' o 'N' |
| 3 | Azienda non apribile in uno o più anni indicati come parametro.<br>Export non effettuato |
| 4 | Parametro 'Calcolo margine' errato o assente. Valori accettati 'M'- movimento o 'A'- anagrafica articoli |
| 5 | Parametro 'Ordine cliente' errato o assente. Valori accettati 'S' o 'N' |
| 6 | Parametro 'Ordine fornitore' errato o assente. Valori accettati 'S' o 'N' |
| 7 | Parametro 'Matrici' errato o assente. Valori accettati 'S' o 'N' |
| 8 | Trasferimento o analisi dati in uso dall'utente ??? |
| 9 | Modulo ADCOMMERCIALE non disponibile in questa installazione |

# _ERRAE

### _ERRAE

### _ERRAE$

La _ERRAE e la _ERRAE$ riportano il codice e la descrizione dell'esito della operazione (GET/PUT) sulle variabili _AE? :

| | |
|---|---|
| 0 | esito positivo dell'operazione (nessun messaggio, il campo è vuoto); |
| 1 | errore specificato a _ERRAE$ (PUTAREA) |
| 2 | area non trovata (GETAREA) |
| 3 | parametro area non valorizzato (GETAREA) |

# _ERRAGP

### _ERRAGP

### _ERRAGP$

In Sprix azzera le variabili di struttura archivi _AGP? dell' Agenda di Lavoro.

La _ERRAGP e _ERRAGP$ riportano in Sprix il codice e la descrizione dell'esito dell'operazione (GET/PUT/DEL) sulle variabili di struttura _AGP?

# _ERRAL

### _ERRAL

### _ERRAL$

In Sprix azzera le variabili di struttura archivi _AL? dei Codici Alias.

La _ERRAL e _ERRAL$ riportano in Sprix codice e descrizione dell'esito della operazione (GET/PUT/DEL) sulle variabili _AL? :

| | |
|---|---|
| 0 | esito positivo dell'operazione (nessun messaggio, il campo e vuoto); |
| se <>0 | errore descritto in _ERRAL$; |

# _ERRAR

### _ERRAR

### _ERRAR$

La _ERRAR e la _ERRAR$ riportano in Sprix il codice e la descrizione dell'esito dell'operazione (GET/PUT/DEL) sulle variabili _AR? :

| | |
|---|---|
| 0 | esito positivo dell'operazione (nessun messaggio, il campo è vuoto); |
| 1 | errore generico, descritto in _ERRAR$ (consultarla); |
| 2 | codice articolo errato; |
| 3 | numero magazzino errato; |
| 4 | codice articolo non trovato; |
| 5 | articolo non modificabile (archivio rovinato); |
| 6 | impossibile inserire ulteriori articoli; |
| 7 | articolo non cancellabile (archivio rovinato); |
| 8 | articolo non cancellabile (tipo non ammesso o con progressivi); |
| 9 | articolo non cancellabile (non esistente); |
| 10 | articolo in uso dall'utente...; |
| 11 | articolo: "arcod" ...<br>- non è possibile togliere la struttura;<br>- non è possibile inserire la struttura;<br>- non può essere trasformato in 'A CORPÒ;<br>- non può essere trasformato in non 'A CORPÒ;<br>- non può essere trasformato in a taglie;<br>- deve essere obbligatoriamente una taglia;<br>- non può essere trasformato in a peso netto;<br>- deve essere obbligatoriamente a peso netto; |

# _ERRAZIE

### _ERRAZIE

### _ERRAZIE$

APRIAZIE e APRIAZIE_EXT impostano il codice e descrizione dell'esito dell'operazione _ERRAZIE e _ERRAZIE$):

| | |
|---|---|
| 0 | operazione eseguita con successo nessun messaggio, il campo è vuoto |
| 1 | alcuni file dell'azienda sono mancanti; |
| 2 | l'azienda è di livello superiore al dispositivo di protezione; |
| 3 | la sigla e/o la data non sono validi; |
| 4 | l'azienda non esiste; |
| 5 | la data non è gestita: non esistono archivi in linea per questa data; |
| 6 | gli anni aziendali sono errati: il file AZIE non è valido; |
| 7 | l'accesso all'azienda non è consentito; |
| 8 | troppe aziende installate; |
| 9 | azienda non gestibile; |
| 99 | azienda non apribile (errore generico); |

# _ERRBL

### _ERRBL

### _ERRBL$

In Sprix/Collage azzera le variabili di struttura archivi _OB? e _OL? della Bolla di Lavorazione.

La _ERRBL e la _ERRBL$ riportano in Sprix il codice e la descrizione dell'esito dell'operazione (GET/PUT/DEL) sulle _OB? e _OL? :

| | |
|---|---|
| 0 | esito positivo dell'operazione (nessun messaggio, il campo è vuoto); |
| 1 | errore generico con relativa descrizione;<br>- riga OC non valida;<br>- riga già collegata a XX xx/xxxxxx;<br>- l'articolo ???? è senza DBP;<br>- codici incompatibili: '????' e '????';<br>- incompatibilità tipo articolo/quantità;<br>- documento in uso da altro terminale riga non modificabile;<br>- riga già variata da altro terminale; |

| | |
|---|---|
| | - riga con distinta base automatica riga non modificabile;<br>- riga bloccata riga non modificabile;<br>- riga ordine cliente annullata riga non modificabile;<br>- errore generico riga non modificabile; |
| 2 | numero bolla errato; |
| 3 | memoria insufficiente; |
| 4 | bolla padre non trovata; |
| 5 | bolla non trovata; |
| 6 | errore generico; |
| 7 | numero BL padre diverso da quello effettivo; |
| 8 | non è possibile inserire una sottobolla; |
| 9 | bolla Padre non esistente; |
| 10 | bolla non totalmente cancellata; alcune righe OC non scollegabili; |
| 11 | documento già in uso dal terminale ??; |
| 12 | art.???? documento OC collegato già in uso; |
| 13 | art.???? collegamento con riga OC non valido; |
| 14 | art.???? Non è possibile collegare righe OC; |
| 15 | art.???? Riferimento OC non ammesso, su produzione per il magazzino; |

# _ERRCDG

### _ERRCDG

### _ERRCDG$

riportano il codice e descrizione dell'esito dell'operazione (GET/PUT/DEL) sulle variabili _CDG? :

| | |
|---|---|
| 0 | esito positivo dell'operazione (nessun messaggio, il campo è vuoto); |
| 1 | parametri istruzione errati; |
| 2 | anno errato |
| 3 | errore elaborazione istruzione, documento non trovato. |
| 4 | errore in GETCDG: codifica errata parametro chiave KeyDoc. |
| 5 | errore in PUTCDG: documento in uso da un altro terminale. |

# _ERRCO

### _ERRCO

### _ERRCO$

riportano il codice e la descrizione dell'esito dell'operazione (GET/PUT) sulle variabili _CO? :

| | |
|---|---|
| 0 | esito positivo dell'operazione (nessun messaggio, il campo è vuoto); |
| 1 | errore specificato in ERRCO$ (PUTCOMM) |
| 2 | commessa non trovata (GETCOMM) |
| 3 | parametro commessa non valorizzato (GETCOMM) |

# _ERRCOM

### _ERRCOM$

In Sprix/Collage riporta la descrizione dell'esito dell'operazione di gestione della porta seriale: "" (messaggio vuoto, assenza di errore); "Errore di apertura porta" (errore open porta, consigliato il reset macchina); "Errore lettura settaggio porta" (errore di porta "sporca", tipico della precedente chiusura anomala, consigliato il reset macchina); "Errore settaggio porta" (errore nella fase open indipendente dai parametri, consigliato il reset macchina); "EscapeCommFunction" (errore di inizializzazione della seriale in fase di apertura, consigliato reset macchina); "Errore lettura parametri dalla porta" (errore lettura dei parametri preimpostati sulla seriale, consigliato il reset macchina); "Byte size errato: valori possibili: 5,6,7,8" (è stata specificata una dimensione della parola errata, passare esclusivamente uno dei valori indicati); "Parità sbagliata: valori possibili: Nessuna, Pari, Dispari" (è stato indicato un valore di parità diverso da quelli ammessi: 0,1,2); "Baud Rate errato: valori possibili 110,300,600,1200,2400,4800,9600,19200,38400" (è stato indicato un valore di Baud Rate errato, passare solo uno dei valori indicati); "Errore settaggio parametri della porta" (errore dovuto alle caratteristiche dell'hardware: i parametri passati non sono compatibili con il tipo di porta seriale); "Errore chiusura porta" (errore che si può ricevere su una successiva fase di lettura e che potrebbe dipendere anche dall'utilizzo della seriale da parte di altre applicazioni).

```
Esempio Sprix
APRI:
OPENCOM "COM2",9600,2,"7",0,60 : IF _ERRCOM$<>0 THEN GOTO ERRORE
DIM C$(): DIM Q(): EXIT=0: P=0: J=1
LEGGI_TESTA:
 GETCOM 1 : IF _ERRCOM$<>0 THEN GOTO ERRORE
V=ASC(_BUFCOM$): IF V<>ASC("H") THEN GOTO LEGGI_TESTA
LEGGI:
FOR I=1 TO 80 : GETCOM 1:  IF _ERRCOM$<>0 THEN GOTO ERRORE
IF _BUFCOM$="" THEN EXIT=1
IF _BUFCOM$="[" THEN GOTO COMP
IF EXIT=1 THEN GOTO STOP
NEXT
COMP:
FOR I=1 TO 80 : GETCOM 1 : IF _ERRCOM$<>0 THEN GOTO ERRORE
```

```
V=ASC(_BUFCOM$)
IF V=ASC("]") OR V=ASC("H") THEN C$(J)=LEFT$(C$(J),LEN(C$(J))-1):GOTO FINE
C$(J)=C$(J)+_BUFCOM$
 NEXT
FINE:
 GETCOM 1 : IF _ERRCOM$<>0 THEN GOTO ERRORE
 IF _BUFCOM$="I" THEN GETCOM 1: IF _BUFCOM$ = "N" THEN GOTO STOP
 IF _BUFCOM$=CHR$(13) THEN J=J+1:GOTO LEGGI:ELSE Q(J)=Q(J)+VAL(_BUFCOM$)
 GOTO FINE

' FOR J=1 TO I
'  IF C$(J)=_BUFCOM$ AND P=0 THEN Q(J)=Q(J)+1 ELSE C$(I)=BUFCOM$:Q(I)=1:P=1
' NEXT
'NEXT

STOP:
 FOR X=1 TO J: PRINT C$(X)+"   ";: PRINT STR$(Q(X),0,0): NEXT
CLOSECOM

ERRORE:
 IF _ERRCOM$<> "" THEN VIMSG 5, _ERRCOM$: END

 OPEN....
  .... VAR$="Sprix"
 PUTCOM VAR$,5 : IF _ERRCOM$<>0 THEN GOTO ERRORE
  CLOSECOM
```

# _ERRDA

### _ERRDA

### _ERRDA$

In Sprix/Collage azzera le variabili di struttura archivi _DA? di Anagrafica Indirizzi di Spedizione.

La _ERRDA e _ERRDA$ riportano in Sprix il codice e la descrizione dell'esito dell'operazione (GET/PUT) sulle _DA? :

| | |
|---|---|
| 0 | esito positivo dell'operazione |
| 1 | dati immessi in modo errato (scrittura) |
| 2 | codice non trovato (lettura) |
| 3 | ultima anagrafica già utilizzata (scrittura) |

# _ERRDB

### _ERRDB

### _ERRDB$

In Sprix azzera tutte le variabili di struttura archivi _TP? della Distinta Base Primaria.

_ERRDB e _ERRDB$ riportano in Sprix il codice e descrizione dell'esito della operazione (GET/PUT/DEL) sulle variabili _TP? :

| | |
|---|---|
| 0 | esito positivo dell'operazione (nessun messaggio, il campo è vuoto); |
| 1 | errore generico descritto in _ERRDB$; |
| 2 | distinta base non trovata; |
| 3 | codice distinta base non valido; |
| 4 | num. fase non valido, il num. fase che si vuole leggere/scrivere è>99; |
| 5 | numero fase non valido; |

# _ERRDBMIRR

### _ERRDBMIRR$

Le ASCMIRROR_EXT, ASCMIRROR_EXT2, DBMIRROR, DBMIRROR_EXT valorizzano la _ERRDBMIRR$ che rilasciano la descrizione dell'esito dell'operazione effettuata (ulteriori registrazioni avvengono anche nel file di log pxerrsql.<num term.> per ulteriori informazioni consultare nella directory del disco della versione "gest\doc\conces\dbmirror\"):

# _ERRDBMIRROR

Le ASCMIRROR_EXT, ASCMIRROR_EXT2, DBMIRROR, DBMIRROR_EXT valorizzano le _ERRDBMIRROR che rilascia il codice dell'esito dell'operazione effettuata (ulteriori registrazioni avvengono anche nel file di log pxerrsql.<num term.> per ulteriori informazioni consultare nella directory del disco della versione "gest\doc\conces\dbmirror\"):

| | |
|---|---|
| -1 | elaborazione conclusa correttamente, ma con presenza di uno o più errori non bloccanti (WARNING); |
| 0 | esito positivo dell'operazione (nessun messaggio, il campo è vuoto); |
| 1 | errore durante la connessione al database, verificare il profilo DSN; |
| 2 | database non supportato; |
| 3 | errore ODBC sulla funzione GetSQLInfo; per DBMIRROR_EXT, non sono disponibili le informazioni inerenti l'RDBMS; |
| 4 | errore generico su ODBC; |
| 5 | tabella non gestita; per DBMIRROR_EXT, tabella <riferimento> non trovata nel file risorse; |
| 6 | tabella non ancora implementata; per DBMIRROR_EXT, tabella <riferimento> non gestita dalla DBMIRROR; |
| 7 | archivio sprix non trovato; per DBMIRROR_EXT, file <riferimento> non disponibile; |
| 8 | tipo dato DB <nome colonna RDBMS> differente da tipo dato sprix <voce dizionario>; |
| 9 | campo sprix <voce dizionario> non trovato; verificare l'integrità del file risorse; |
| 10 | dimensione del campo DB <nome colonna> inferiore alla dimensione del campo sprix <voce dizionario>; |
| 11 | errore sul file di risorsa "dbmirror.dat"; per DBMIRROR_EXT,non trovato<nome file risorse>; |
| 12 | directory <nome directory> di export file ASCII non trovata; |
| 13 | elaborazione interrotta dall'utente; |
| 14 | file di personalizzazione <nome file> non trovato; |
| 15 | colonna <nome colonna> indicata in file di personalizzazione non valida |

Esempio Sprix:

```
_INTER=1: ASCMIRROR "AR_ANAGRAFICHE","",""
IF _ERRASCMIR THEN VIMSG 5, STR$(_ERRASCMIR,0,0): END
_INTER=1: LEGGIDESC=1: DBMIRROR "MM_RIGHE"
IF _ERRDBMIRROR THEN VIMSG 5, _ERRDBMIRR$: END
ASCMIRROR_EXT2 0,"AR_ANAGRAFICHE","C:\a","artic.txt",0,0,0,0,"SV=1,CI=;CT=""
```

# _ERRDC

### _ERRDC

### _ERRDC$

In Sprix/Collage azzera le variabili di struttura archivi _DC? della Anagrafica Contatti.

La _ERRDC e la _ERRDC$ riportano in Sprix il codice e la descrizione dell'esito dell'operazione (GET/PUT/DEL) sulle _DC? :

| | |
|---|---|
| 0 | esito positivo dell'operazione, nessun messaggio |
| <> 0 | errore, con relativa descrizione |

# _ERRDV

### _ERRDV

### _ERRDV$

In Sprix azzera le variabili di struttura archivi _DV? dei Documenti Docuvision.

La _ERRDV e la _ERRDV$ riportano in Sprix il codice e la descrizione dell'esito dell'operazione (GET/PUT/DEL) sulle _DV? :

| | |
|---|---|
| 0 | Esito positivo dell'operazione, nessun messaggio, il campo è vuoto); |
| 1 | Errore generico con relativa descrizione; |
| 2 | Funzione non eseguibile; |
| 3 | Documento non trovato; |
| 7 | Parametri errati; |

# _ERRIND

In Sprix/Collage restituisce numero in base all'esito positivo (=0) o negativo (=1) dell'operazione su record.

GETKEY 0=trovato; 1=non trovato

VARKEY 0=variato; 1=non variato, la chiave da variare non esiste o esiste già la nuova chiave

INSKEY 0=inserito; 1=non inserito, esiste già

DELKEY 0= cancellato; 1=non cancellato, non esiste

NEXTKEY 0=prossimo trovato; 1= prossimo non trovato

PREVKEY 0=precedente trovato; 1=precedente non trovato

# _ERRIL

### _ERRIL

### _ERRIL$

In Sprix azzera le variabili di struttura archivi _IL? degli Impegni di Lavorazione.

La _ERRIL e la _ERRIL$ riportano in Sprix il codice e descrizione dell'esito dell'operazione (GET/PUT/DEL) sulle _IL???:

| | |
|---|---|
| 0 | esito positivo dell'operazione; nessun messaggio, il campo è vuoto); |
| 1 | errore generico (solo PUTIL); |
| 1 | codice art. materie prime non trovato nel file articoli (solo GETIL); |
| 2 | impegni non trovati in archivio (solo GETIL); |

## _ERRLOT

### _ERRLOT

### _ERRLOT$

In Sprix azzera le variabili di struttura archivi _LOT? di Anagrafica Lotto.

La _ERRLOT e _ERRLOT$ riportano in Sprix il codice e la descrizione dell'esito dell'operazione (GET/PUT/DEL) sulle _LOT? :

| | |
|---|---|
| 0 | esito positivo dell'operazione (nessuna descrizione, messaggio vuoto); |
| 1 | errore generico di congruità dati;<br>- "Tipo dato personalizz. errato.Cod.<TpLotto><CodDatoUtente>(lotto)"<br>- "Valore dato personalizz. errato.Cod.<TpLotto><CodDatoUtente>(lotto)"<br>- "Valore dato personal.fuori limiti.Cod.<TpLotto><CodDatoUtente>(lot)"<br>- "Dato person. contat. non modificab.Cod.<TpLotto><CodDatoUtente>(lot)"<br>- "Dato pers.File nome errato/inesist. Cod.<TpLotto><CodDatoUtente>(lot)"<br>- "Dati personalizzati non gestibili (lotto )" |
| 2 | errore dato utente obbligatorio (in fase di scrittura);<br>- "Dato personalizzato obbligat. Cod. <TpLotto><CodDatoUtente> (lotto)" |
| 3 | errore lettura lotto<br>- "Codice lotto non trovato (lotto)"<br>- "Codice lotto non valido o non trovato lotto ()" |
| 4 | errore lotto in uso da altri termin. "Lotto in uso da utente()(lotto)" |
| 5 | errore cancellaz. lotto "Codice lotto movimentato non cancellab.(lot)" |

## _ERRMYDB

### _ERRMYDB

### _ERRMYDB$

In Sprix/Collage restituisce numero/descrizione in base all'esito positivo (=0) o negativo (=1) dell'operazione.

MYDBGET (0=trovato; 1=non trovato),

MYDBPUT (0=inserito/Variato; 1=non inserito/variato, esiste già),

MYDBDEL (0= cancellato; 1=non cancellato, non esiste),

MYDBNEXT (0=prossimo trovato; 1= prossimo non trovato),

MYDBPREV (0=precedente trovato; 1=preced. non trovato).

| | |
|---|---|
| 101 | Errore generico _ERRMYDB$ contiene il dettaglio dell'errore |
| 102 | Archivio MyDB non definito 'Prod_Cod' |
| 103 | Numero Key non valida per archivio MyDB ' Prod_Cod' |
| 104 | Istruzione ****** non eseguita |
| 105 | ID (###) archivio MyDB ' Prod_Cod ' in uso da un altro terminale |
| 106 | _MYDBID non impostato archivio MyDB ' Prod_Cod' |
| 107 | Numero finestra errato |
| 108 | Dato non modificabile (Ritornato in caso di tentativo di modifica di un record Bloccato Ctrl+F3) |
| 109 | ID (###) archivio MyDB ' Prod_Cod ' gia' in uso dal terminale |
| 110 | Archivio MyDB non trovato 'Prod_Cod' |
| 110 | Record non inserito (Riorganizzare 'Prod_Cod') |
| 111 | Record non variato (Riorganizzare 'Prod_Cod') |
| 112 | Record non cancellato (Riorganizzare 'Prod_Cod') |

## _ERRMM

### _ERRMM

### _ERRMM$

In Sprix/Collage azzera le variabili di struttura archivi _MM? del documento di magazzino.

La _ERRMM e _ERRMM$ riportano in Sprix il codice e la descrizione dell'esito dell'operazione (GET/PUT/DEL e totalizzazioni) sulle _MM? :

| | |
|---|---|
| 0 | esito positivo dell'operazione (nessun messaggio, il campo è vuoto); |
| 1 | errore generico, la descrizione è in _ERRMM$ (es.: fuori fido); |
| 2 | errore parametri aziendali (conti automatici, fuori fido, ecc...); |
| 3 | errore caricamento documento (non trovato in fase di lettura, ecc...); |
| 4 | documento in uso da altri terminali (in fase di scrittura documento); |
| 5 | documento da cancellare non trovato; |
| 6 | la cancellazione dell'operazione di primanota non è possibile; |
| 7 | parametro non valido; |
| 10 | Uguale a 0 ma relativo a beni usati; |

| 11 | Uguale a 1 ma relativo a beni usati; |
|---|---|
| 12 | Uguale a 2 ma relativo a beni usati; |
| 13 | Uguale a 3 ma relativo a beni usati; |

messaggi relativi alla variabile _MMNRECRIGA

DBA con troppi componenti

DBA con riferimenti incrociati o memoria insufficiente

La lista dei componenti potrebbe essere incompleta (riga <Numero riga>)

Uno o più componenti hanno superato la q.tà massima (riga <Numero riga>)

# _ERRPART

### _ERRPART

### _ERRPART$

In Sprix/Collage azzera le variabili di struttura archivi _PART? delle Particolarità Conti e Articoli.
La _ERRPART e la _ERRPART$ riportano in Sprix il codice e descrizione dell'esito dell'operazione (GET/PUT/DEL)sulle _PART? :

| 0 | esito positivo dell'operazione (nessun messaggio, il campo è vuoto); |
|---|---|
| 1 | errore generico descritto in _ERRPART$; |
| 2 | errore tipo particolarità non valido (valori ammessi P/S/V/A); |
| 3 | occorre indicare almeno uno tra la categoria ed il codice cliente; |
| 4 | si deve indicare un solo campo tra la categoria ed il codice cliente; |
| 5 | occorre indicare almeno uno tra la categoria ed il codice articolo; |
| 6 | si deve indicare un solo campo tra la categoria ed il codice cliente; |
| 7 | data inizio validità errata, |

# _ERRPC

### _ERRPC

### _ERRPC$

riportano in Sprix il codice e la descrizione dell'esito dell'operazione (GET/PUT/DEL) sulle variabili _PC? :

| 0 | esito positivo dell'operazione (nessun messaggio, il campo è vuoto); |
|---|---|
| 1 | errore generico, descritto in _ERRPC$; |
| 2 | codice non trovato nel piano dei conti; |
| 3 | la p.Iva esiste già con un altro codice nello stesso gruppo/mastro; |
| 4 | conto non cancellabile (ultima consegna diversa da zero); |
| 5 | conto non cancellabile (valore ordini/bolle in corso diverso da 0); |
| 6 | conto non cancellabile (progressivi diversi da 0); |

# _ERRPN

### _ERRPN

### _ERRPN$

In Sprix/Collage azzera le variabili di struttura archivi _PN? del documento di primanota.

La _ERRPN e la _ERRPN$ ritornano in Sprix il codice e la descrizione dell'esito dell'operazione (GET/PUT/DEL) sulle variabili _PN? :

| 0 | esito positivo dell'operazione (nessun messaggio, il campo è vuoto); |
|---|---|
| 1 | errore generico. Nella cancellazione i seguenti messaggi:<br>- libro giornale del mese già stampato in maniera definitiva<br>- registro IVA acquisti del mese già stampato in maniera definitiva<br>- registro IVA vendite del mese già stampato in maniera definitiva<br>- registro corrispettivi del mese già stampato in maniera definitiva<br>- liquidazione del mese già stampata in maniera definitiva<br>- registrazione con immobilizzazione già stampata sul registro<br>- registrazione con immobilizzazione in parte o totalmente venduta<br>- registrazione con costo pluriennale già stampato<br>- documento in uso da altro terminale<br>- archivio immobilizzazioni già in uso da un altro terminale<br>- operazione inviata al commercialista |

La cancellazione verifica se il documento (causale+numero+data) esiste in scadenzario: se assente rilascia il messaggio "documento senza record scadenzario, documento non cancellabile"; le manipolazioni dello scadenzario(cancellazioni, pareggi, ...) possono causare la mancanza di questi riferimenti; se questi invece ci sono, la cancellazione può rilasciare i seguenti messaggi: 'nello scadenzario del conto il documento è già in pagamentò (se è già effettuato non consente modifiche e la cancellazione richiede la preventiva rimozione manuale di 'P' dallo scadenzario scollegando la rata del documento); "il documento può essere cancellato solo entrando nel programma con la sua data di registrazione"; "la cancellazione comporta variazione di una registrazione con ritenuta d'acconto già versata".

# _ERRSQL_N

riguarda errori nativi generati dal data base di riferimento;

Non differenziano tra un errore e un Warning (es. all'atto della connect a SQL_Server). La visualizzazione di messaggi prima del test su variabile _ERRSQL potrebbe duplicare record, si consiglia un'apposita label da richiamare.

```
Esempio
GEST_ERR_SQL:
 IF _ERRSQL_N=0 AND _ERRSQL_S$="" THEN RETURN
 --- IF _ERRSQL_M$="" THEN RETURN
 PRINT STR$(_ERRSQL_N,0,0)+" "+_ERRSQL_S$+" "+_ERRSQL_M$
 END
```

# _ERRSQL_M

### _ERRSQL_M$

riguarda errori di istruzioni SQL_ e la lista di errori dipende dal tipo di database (presenti se _ERRSQL_M$<>"") e la sequenza non corretta delle operazioni rilascia errore "function sequence error"; Non differenzia tra un errore e un Warning (es. all'atto della connect a SQL_Server). La visualizzazione di messaggi prima del test su variabile _ERRSQL potrebbe duplicare record, si consiglia un'apposita label da richiamare.

```
Esempio
GEST_ERR_SQL:
 IF _ERRSQL_N=0 AND _ERRSQL_S$="" THEN RETURN
 --- IF _ERRSQL_M$="" THEN RETURN
 PRINT STR$(_ERRSQL_N,0,0)+" "+_ERRSQL_S$+" "+_ERRSQL_M$
 END
```

# _ERRSQL_S

### _ERRSQL_S$

riguarda errori interni di ODBC (connessione, ecc.) e rappresenta il valore di SQL State (ulteriori informazioni sul manuale ODBC). Non differenziano tra un errore e un Warning (es. all'atto della connect a SQL_Server). La visualizzazione di messaggi prima del test su variabile _ERRSQL potrebbe duplicare record, si consiglia un'apposita label da richiamare.

```
Esempio
GEST_ERR_SQL:
 IF _ERRSQL_N=0 AND _ERRSQL_S$="" THEN RETURN
 --- IF _ERRSQL_M$="" THEN RETURN
 PRINT STR$(_ERRSQL_N,0,0)+" "+_ERRSQL_S$+" "+_ERRSQL_M$
 END
```

# _ERRTPLOT

### _ERRTPLOT

### _ERRTPLOT$

In Sprix azzera le variabili di struttura archivi _TPLOT? di Anagrafica Tipo Lotto.

La _ERRTPLOT e la _ERRTPLOT$ riportano in Sprix il codice e la descrizione  dell'esito dell’operazione GET/PUT/DEL sulle _TPLOT?:

| 0 | esito positivo dell'operazione (nessuna descrizione, messaggio vuoto); |
|---|---|
| 1 | errore generico di congruità dati: "Cod. tipo lotto obbligatorio.()"; |
| 2 | errore dato utente non cancellabile (in scrittura):<br>- "Cancellazione dato pers. non ammessa. Cod. mancante o errato. ()"; |
| 3 | errore lettura tipo lotto:<br>- "Codice tipo lotto non valido o non trovato (<cod tipo lotto>)";<br>- "Codice tipo lotto non valido o non trovato ()";<br>- "Codice tipo lotto non immesso (<cod tipo lotto>)";<br>- "Tipo ordinamento non valido. Valori ammessi 1 2 3 (<cod tipo lotto>)"; |
| 4 | errore tipo lotto in uso da terminali: "Tipo lotto in uso da utente()"; |
| 5 | errore cancellazione tipo lotto: "<br>- Codice tipo lotto non cancellabile. Stato diverso da 'C'()" |

Ulteriori controlli:

- Controlli sui valori ammessi riportati in tabella
- Cod. tipo lotto su dati utente, tutti uguali
- Cod. dato utente obbligatorio se impostata la parte del codice tipo lotto
- Se tipo dato utente è
- A controllo valore default con dimensione impostata (da 1 a 80)
- N ctrl.val.def. con dimensione impostata (da 1 a 10)
- I ctrl.dim.impost.(da 1 a 9); val.def. non ammesso; ctrl. singolo dato I
- V ctrl.val. default con dim. impostata (da 1 a 15)
- D ctrl.val.def. con data valida o dicitura SISTEMA o TERMINALE
- F ctrl.val.def. con dim. impost. (da 1 a 255) e ctrl. esist. file immesso
- Controllo univocità Ordine tra i dati utente
- Ctrl. univocità elemento cod. utente (chiave) tra i dati personalizzati e se chiave il dato deve essere obbligatorio

# _GLOBAL

## _GLOBAL()

## _GLOBAL$()

In Collage sono array globali (numerico e stringa) di massimo 32767 elementi ciascuno dove memorizzare dati generati da un Collage per renderli disponibili contemporaneamente a tutti i Collage caricati in memoria (in modo AUTO su evento o con CALL o STARTCOLLAGE), con visibilità/comunicabilità reciproca fra valori delle variabili e le routine su etichetta di Collage diversi.

# _IL

variabili di struttura archivi Impegni di Lavorazione (GET-PUT-DEL)
In rosso evidenziati i campi minimi obbligatori

| | |
|---|---|
| _ILNDO(_NIL) | (numero documento) |
| _ILNSD(_NIL) | (numero sottodocumento) |
| _ILNRB(_NIL) | (num. progr. riga bolla; -1="RES_SL", -2="RES_DL") |
| _ILFAS(_NIL) | (numero fase prod. finito) |
| _ILMAG(_NIL) | (magazzino materie prime) |
| _ILDAD$(_NIL) | (data scadenza riga bolla) |
| _ILAMP$(_NIL) | (cod. art. Mat. Prima) |
| _ILTQT(_NIL) | (1-99 taglie; 101 tipo art um1; 102 tipo art um2) |
| _ILCOL(_NIL) | (num. colli o qta variabile) |
| _ILQTA(_NIL) | (quantità o qta fissa) |
| _ILQTG(_NIL) | (quantità per taglia) |
| _ILQTGXL(_NIL) | (quantità per taglia da 17 a 32) |
| _ILPRZ(_NIL) | (prezzo) |
| _ILEVI$(_NIL) | (evidenzia; ' ','È','L','T') |
| _ILIND$(_NIL) | (indice) |
| _ILDEP(_NIL) | (1 depositato, 0 non depositato) |
| _ILEDL(_NIL) | (1 = si, 0 = no) |
| _ILESL(_NIL) | (1 = si, 0 = no) |
| _ILLAV(_NIL) | (1 = si, 0 = no) |
| _ILFOR$(_NIL) | (codice fornitore) |

Esempio:
legge gli Impegni di Lavorazione di una BL in base al numero di riferimento PF, visualizza i dati di una Materia Prima con ciclo di lettura in base _NIL e varia la quantità della MP al Rif. PF (n.1)

```
GETIL 19,0,1,1,0,"MP-ORE" : IF _ERRIL<>0 THEN GOTO ERRORE
FOR I=1 TO _NIL
LPRINT "NDOC.: "; : LPRINT _ILNDO(I)
LPRINT "NSOT.: "; : LPRINT _ILNSD(I)
LPRINT "NROPF: "; : LPRINT _ILNRB(I)
LPRINT "FASE.: "; : LPRINT _ILFAS(I)
LPRINT "MAG..: "; : LPRINT _ILMAG(I)
LPRINT "CODMP: "; : LPRINT _ILAMP$(I)
NEXT
IF _ILQTA(1)>2 THEN _ILQTA(1)=2.2
PUTIL : IF _ERRIL<>0 THEN GOTO ERRORE
ERRORE: IF _ERRIL<>0 THEN VIMSG 4, _ERRIL$
END
```

# _INTER

In Sprix/Collage attiva (se =1, default) o disattiva (se =0) la richiesta di interruzione forzata del programma con tasto Esc/Fine (equivalente a Ctrl+C)

Se =0 il tasto Esc/Fine è attivo se impostato con SETTF.

# _INTEST

In Sprix/Collage abilita (se =1, default) o disabilita (se =0 e ripristinabile a = 1) la stampa delle intestazioni. Elimina la formattazione della stampante con intestazioni (es. righe 'H'); non ha effetto nelle stampe senza la specifica di etichette di intestazione.

# _LOT

variabili di struttura archivi Anagrafica Lotti (GET-PUT-DEL)

| | |
|---|---|
| _LOTID | (ID lotto); |
| _LOTSTAT$ | (Stato lotto); |
| _LOTMESG$ | (Messaggio stato bloccato/annullato); |
| _LOTCARM$ | (Carico effettuato da mov. magazz. s/n); |
| _LOTTCCA$ | (Tipo Controllo carico lotto); |

| | |
|---|---|
| _LOTTCCANEW$ | (Controllo carico memorizzato in anagrafica lotto: se impostata modifica il tipo controllo: da M può andare a F o D o A; da Do F può andare ad A; se impostato un nuovo metodo di gestione dell'anagrafica lotto non èpossibile tornare alla precedente) |
| _LOTCART$ | (Codice articolo); |
| _LOTCTPL$ | (Codice tipo lotto); |
| _LOTCFOR$ | (Codice fornitore); |
| _LOTDTCA$ | (Data di carico); |
| _LOTDTVA$ | (Data validità); |
| _LOTDTCR$ | (Data creazione) Sola Lettura; |
| _LOTDTUM$ | (Data ultima modifica) Sola Lettura; |
| _LOTCUTE$ | (Codice utente lotto); |
| _LOTDUCD$ (999) | (Codice campo dato utente personalizz.); |
| _LOTDUTP$ (999) | (Tipo dato utente personalizzato); |
| _LOTDUVL$ (999) | (Valore dato utente personalizzato); |

Esempio: Scrive lotto per articolo "04M-MI", lo legge e blocca e lo rilegge con GETMULTILOT. Infine lo cancella.

```
_LOTSTAT$="C" : _LOTCART$="04M-MI" : _LOTCTPL$="ab"
 _LOTCFOR$="401.00001" : _LOTDTCA$="20210101" : _LOTDTVA$="20210101"
 _LOTDUCD$(1)="xx" : _LOTDUTP$(1)="A" : _LOTDUVL$(1)="For.1"
 _LOTDUCD$(2)="yy" : _LOTDUTP$(2)="A" : _LOTDUVL$(2)="Cli.2"
 _LOTDUCD$(3)="zz" : _LOTDUTP$(3)="A" : _LOTDUVL$(3)="Serie 3"

PUTLOT: IF _ERRLOT<>0 THEN VIMSG 4, _ERRLOT$
 VIMSG 1, "Ha registrato il lotto "+STR$(_LOTID,0,0)+" per "+_LOTCART$
 IDLOT=_LOTID
 AZZVARSYS 17

GETLOT IDLOT : IF _ERRLOT<>0 THEN VIMSG 1, _ERRLOT$
 _LOTSTAT$="B":_LOTMESG$="messaggio per il blocco"
PUTLOT: IF _ERRLOT<>0 THEN VIMSG 4, _ERRLOT$
 VIMSG 1, "Ha bloccato il lotto "+STR$(_LOTID,0,0)+" di "+_LOTCART$

GETMULTILOT "","04M-MI",0,"",1,1 : IF _ERRLOT<>0 THEN VIMSG 1, _ERRLOT$
 FOR I=1 TO _LOTNID_M
  PRINT "Id lotto "+STR$(_LOTID_M(I),0,0)
  PRINT "Codice Tipo lotto "+_LOTCTPL_M$(I)
  PRINT "Codice Articolo "+_LOTCART_M$(I)
  PRINT "Codice Utente lotto "+_LOTCUTE_M$(I)
NEXT
 VIMSG 1, "legge con GETMULTILOT, confermando il messaggio cancella"
DELLOT IDLOT : IF _ERRLOT<>0 THEN VIMSG 1, _ERRLOT$ : END
```

# _LOT_M

variabili di struttura archivi Anagrafica Lotti (GETMULTILOT)

| | |
|---|---|
| _LOTID_M(I) | (ID lotto); |
| _LOTSTAT_M$(I) | (Stato lotto); |
| _LOTMESG_M$(I) | (Messaggio stato bloccato/annullato); |
| _LOTCARM_M$(I) | (Carico effettuato da mov. magazz. s/n); |
| _LOTTCCA_M$(I) | (Tipo Controllo carico lotto); |
| _LOTCART_M$(I) | (Codice articolo); |
| _LOTCTPL_M$(I) | (Codice tipo lotto); |
| _LOTCFOR_M$(I) | (Codice fornitore); |
| _LOTDTCA_M$(I) | (Data di carico); |
| _LOTDTVA_M$(I) | (Data validità); |
| _LOTDTCR_M$(I) | (Data creazione) Sola Lettura; |
| _LOTDTUM_M$(I) | (Data ultima modifica) Sola Lettura; |
| _LOTCUTE_M$(I) | (Codice utente lotto); |

# _MYDB

### _MYDB$()        Dati dell’archivio

Array impostato dalla istruzione MYDBGET, e usato da MYDBPUT

Contiene i dati dell’archivio, ogni campo di definizione si riferisce ad un elemento del vettore. Tutti i valori sono trasformati in stringa.

Le date sono in formato “AAAAMMGG”

# _MYDBANN

### _MYDBANN$

Ritorna “S” se l’anagrafica è annullata logicamente

# _MYDBARC

### _MYDBARC$

Varibaile che ritorna il nome dell’archivio MyDB in cui l’input è posizionato (Per MYDBGETATT, MYDBPUTATT)

# _MYDBCOD

### _MYDBCOD$()

Vettore impostato dall’istruzione MYDBGET **e** MYDBGETATT. Sono ritornati i codici di campo della finestra atttiva

# _MYDBELE

Variabile impostata dall’istruzione MYDBGET ritorna il numero degli elementi MYDB$() impostati

Esempio:

```
MYDBGET "1900074000_interven", 0, 7
APRISTAMP_EXT 127,"Prova",80
FOR I = 1 TO _MYDBELE '_MYDBELE Num degli elementi del vettore _MYDB$()
' _MYDBETIC$ etichetta del campo + _MYDB$(I) valore del campo
IF I = 1 THEN LPRINT LEFT$(_MYDBETIC$(I)+STRING$(30," "),25)+": "+pades(_MYDB$(I)): GOTO
GIRO1
LPRINT LEFT$(_MYDBETIC$(I)+STRING$(30," "),25)+": "+_MYDB$(I):
```

# _MYDBETIC

### _MYDBETIC$()

Variabile impostata dall’istruzione MYDBGET ritorna le etichette dei campi MYDB$() impostati

Esempio:

```
MYDBGET "1900074000_interven", 0, 7
APRISTAMP_EXT 127,"Prova",80
FOR I = 1 TO _MYDBELE '_MYDBELE Num degli elementi del vettore _MYDB$()
' _MYDBETIC$ etichetta del campo + _MYDB$(I) valore del campo
IF I = 1 THEN LPRINT LEFT$(_MYDBETIC$(I)+STRING$(30," "),25)+": "+pades(_MYDB$(I)): GOTO
GIRO1
LPRINT LEFT$(_MYDBETIC$(I)+STRING$(30," "),25)+": "+_MYDB$(I):
```

# _MYDBF

### _MYDBF$()

Array utilizzato dall’istruzione MYDBVIEW, azzerato dal comando AZZVARSYS(35)

Il funzionamento è identico all’array _MYDB$() della tabella che si vuole filtrare.

Ogni campo di definizione della tabella si riferisce ad un elemento del vettore. Tutti i valori sono stringa.

Se l’elemento del vettore è una data, è presente anche il campo _MYDBF_DTA$() per poter fare delle selezioni per periodo (Da data → A data). Se si vuole selezionare una data sola è comunque necessario inserire la stessa data su entrambi i campi.

Se l’elemento del vettore è un campo alfanumerico o fa riferimento ad un’anagrafica MyDB è presente anche il campo _MYDBF_TP$() che attiva la ricerca avanzata, con la possibilità di specificare il tipo di controllo da effettuare sul campo:

| | |
|---|---|
| * | Ricerca (NON controlla maiuscolo/minuscolo) |
| $ | Ricerca (Controlla maiuscolo/minuscolo) |
| = | Uguale (Controlla maiuscolo/minuscolo) |
| | Uguale (Ricerca classica - NON controlla maiuscolo/minuscolo) |
| . | Inizia per (Controlla maiuscolo/minuscolo) |
| , | Inizia per (NON controlla maiuscolo/minuscolo) |

La MYDBVIEW visualizza la tabella filtrata , in base ad ogni elemento del vettore valorizzato.

***NOTA BENE:***

*Nel caso in cui l’elemento dell’array _MYDBF$() sia un campo dell’archivio stesso, si imposta il valore del campo.*

*Se l’elemento dell’array _MYDBF$() è una relazione ad un altro archivio o tabella MYDB, il valore da impostare nell’elemento _MYDBF$(X) è il MYDBID dell’archivio relazionato.*

# _MYDBK

### _MYDBK$()

Variabile usata dall'istruzione MYDBGET per indicare i vari elementi della chiave.

E' attivo solo il primo elemento _MYDBK$(1) è fisso.

Esempio:
Indica che la chiave da utilizzare nelle relative istruzione è quella del campo impostato come ordinamento 1
_MYDBK$(1) = "501.00001"
MYDBGET "1900074000_interven", 1, 0
IF _ERRMYDB$ <> "" THEN VIMSG 5,_ERRMYDB$
****Ricercare tramite l'estensione**
Esempio:
Indica che la chiave da utilizzare nelle relative istruzione è quella del campo impostato come estensione (99)
_MYDBK$(1) = ”501.00001”
MYDBGET "1900074000_interven", 99, 0
IF _ERRMYDB$ <> "" THEN VIMSG 5,_ERRMYDB$

# _MYDBID

Numero ID anagrafica corrente. Impostata/usata da _MYDBGET, _MYDBPUT, _MYDBPREV e _MYDBNEXT

# _MYDBNR

Numero record dell'anagrafica corrente. Impostata da _MYDBGET, _MYDBPUT, _MYDBDEL, _MYDBPREV, _MYDBNEXT e _MYDBVIEW

# _MYDBTF

Nelle instruzioni MYDB ritorna il codice del tasto funzione utilizzato.

# _MM

Variabili struttura archivi Documenti Magazzino (GET-PUT-DEL-totali) (R) = righe documento; (T) = riferimenti testate (fusione documenti) (L) = riga lotto (max 200)

In rosso evidenziati i campi minimi obbligatori

| TESTATA DOCUMENTO | |
|---|---|
| _MMSIG$ | Sigla Documento |
| _MMSAZ | Codice sottoazienda doc.attuale, v. APRIAZIE |
| _MMMODSTP$ | Tipo modulo di stampa |
| _MMCMO(T) | Causale movimento |
| _MMSER | Serie documento; se il destinatario del documento gestisce la fattura elettronica o la fattura PA (Pubblica Amministrazione), il numero di serie è prelevato in automatico |
| _MMNUM | Num.doc.; 0=AUTO, dopo PUT doc.nuovo contiene il numero *b |
| _MMDAT$ | Data documento; valorizzata al ON_DELMM: |
| _MMDATVAR$ | Data ultima variazione documento; solo tecnologia BP |
| _MMMAG | Magazz.; v. anche Magazz.riga |
| _MMAMA | A Magaz; se 0 solo scar. _MMMAG; v. anche A Magaz.riga |
| _MMCCR | Centro di costo/ricavo |
| _MMCLI$ | Codice cliente/fornitore |
| _MMCODCONT | Codice contatto |
| _MMSTA(T) | Cat.stat. cliente/forn. |
| _MMZON(T) | Zona cliente/forn. |
| _MMIEV$ | Sped.doc.inter. evadibile S/N |
| _MMALISPTRA$ | Aliquota IVA ripartizione spese trasporto |
| _MMEVD$ | Nuova riga evad.di default S/N |
| _MMAGE$ | Codice agente - Tipo F/P |
| _MMAGEDEF$ | Se=1 -> MMAGE$ è default -non del doc.- sola lettura |
| _MMAGECOND | Numero condizione agente |
| _MMNOT$(T) | Nota |
| _MMDOCOLD | Se=1 -> doc. ante v12.3 -arr.dif.-; se=0 post -arr.med- |
| _MMMRA$(T,12) | stato doc. attivo S/N |
| _MMMRV$(T,12) | stato doc. variato S/N; sola lettura |
| _MMFFDP$ | Indica se FF IntraCee a doppio protcollo -sola lettura- |
| _MMEST | Valuta documento |
| _MMVAL | Valuta - Cambio Valuta Doc. -> Valuta Gestione *a |
| _MMCVF$ | Valuta - Cambio; 'V'=variabile, 'F'=fisso |
| _MMVVL$ | Valuta - Fissa prezzi; 'C'=Val.Contab., 'D"=Val.Doc. |
| _MMEVE+_MMDIE | Cambio valuta Doc.->Euro *a+ Flag cambio 0=No/1=Si *a |

| _MMEVC+_MMDIC | Cambio Euro->valuta Gest *a + Flag cambio 0=No/1=Si *a |
|---|---|
| **Riferimenti Documenti** | |
| _MMRBO | Rif.Doc. - Nr.rif. bolla di lav. *b |
| _MMRSB | Rif.Doc. - Nr.rif. sottobolla di lav. *b |
| _MMRFB | Rif.Doc. - Nr.rif. fase di lavorazione *b |
| _MMMSAZ(T) | Rif.Doc. - cod.sottoaz. doc.mov.mag., v.APRIAZIE *b |
| _MMOSAZ(T) | Rif.Doc. - cod.sottoaz. doc.ord.orig., v.APRIAZIE *b |
| _MMOSI$(T) | Rif.Doc. - sigla doc. ordine *b |
| _MMOSE(T) | Rif.Doc. - serie doc. ordine *b |
| _MMONU(T) | Rif.Doc. - numero doc. ordine *b |
| _MMODA$(T) | Rif.Doc. - data doc. ordine )*b |
| _MMMSI$(T) | Rif.Doc. - sigla doc. origine mov.mag. *b |
| _MMMSE(T) | Rif.Doc. - serie doc. origine mov.mag. *b |
| _MMMNU(T) | Rif.Doc. - numero doc. origine mov.mag. *b |
| _MMMDA$(T) | Rif.Doc. - data doc. origine mov.mag. *b |
| _MMSIGRE$ | Rif.Doc.Est. - Sigla doc.attuale *b |
| _MMNUMRE$ | Rif.Doc.Est. - Numero doc.attuale *b |
| _MMDATRE$ | Rif.Doc.Est. - Data doc.attuale *b |
| _MMMSIRE$(T) | Rif.Doc.Est. - Sigla doc.di car/scar *b |
| _MMMNURE$(T) | Rif.Doc.Est. - Num doc.di car/scar *b |
| _MMMDARE$(T) | Rif.Doc.Est. - Data doc.di car/scar *b |
| _MMOSIRE$(T) | Rif.Doc.Est. - Sigla doc.ord.origine *b |
| _MMONURE$(T) | Rif.Doc.Est. - Num doc.ord.origine *b |
| _MMODARE$(T) | Rif.Doc.Est. - Data doc.ord.origine *b |
| **Pubblica Amministrazione** | |
| _MMRIFAMMPA$ | Riferimento amministrazione PA |
| _MMRIFFTPA$ | Riferimento numero fattura collegata PA |
| _MMRIFDATA$ | riferimento data fattura collegata PA |
| _MMTPRIFRE$ | Tipo Identificativo ai riferimenti esterni (FPA)<br>“ ”=Documento(default),C=Contratto,V=Convenzione |
| _MMCTCVRE$ | Codice del documento per tipo identificativo “C” o “V” (FPA) |
| _MMMTPRIFRE$(T) | Tipo Identificativo ai riferimenti esterni (FPA) derivamente da documento origine |
| _MMMCTCVRE$(T) | Identificativo documento origine (FPA) derivamente da documento origine |
| _MMOTPRIFRE$(T) | Tipo Identificativo ai riferimenti esterni (FPA) derivante da ordine cliente |
| _MMOCTCVRE$(T) | Identificativo documento origine(FPA) derivante da ordine cliente |
| _MMALICAPRO$ | Aliquota IVA cassa professionista - Az. prof. -<br>(aliquota in GET/PUT; salvata nel documento di ultimo livello i documenti emessi precedentemente usano aliquota di tabella,quelli con data < 15-9-2011 usano il 20%; in revisione se è impostata = 0 legge la tabella) |
| _MMPERCAPRO$ | Percentuale cassa professionista - Az. prof. - |
| _MMALIINPS$ | Aliquita Iva Inps |
| _MMPERINPS | Percentuale Inps |
| _MMTPSPTRO$ | Operazione riferita a contratto S/N |
| _MMCROSPTRO$ | Codice Contratto |
| _MMCIMMCORI$ | CO/RI contabilizzato primanota S/N -solo revisione |
| _MMCIC$() | Cointestatari spesometro - conto |
| _MMCIP() | Cointestatari spesometro - percentuale |
| | *Parametri documenti BD,DL,SL,CL: per doc. nuovo sono presi i valori da tabella Parametri Documenti, per il doc. già esistente in archivio la PUTMM utilizza le variabili memorizzate in fase di creazione del documento.* |
| _MMPRGMAG$ | Progressivo Magazzino |
| _MMPRGAMAG$ | Progressivo A Magazzino |
| _MMAGDTMAG$ | Aggiornamento data Magazzino |
| _MMAGDTAMA$ | Aggiornamento data A Magazzino |
| _MMAGGCU$ | Aggiornamento CU -documento CL- |
| _MMGCONDOC$ | Clienti/Fornitori -NCF- |
| **RIGHE DEL DOCUMENTO** | |
| _MMTPR$(R) | Tipo riga articolo D,T,R,r *c |
| _MMNTE(R) | Riferimenti a testate per doc. trasformati |
| _MMART$(R) | Codice articolo -già codificato e leggibile- |
| _MMTSD$(R) | Tipo riga di descrizione '.' o ',' |
| _MMDEE$(R) | Descrizione riga |
| _MMTST$(R) | Tipo scadenza articolo testo |
| _MMARR$(R) | Codice articolo testo |
| _MMCAU(16) | Quantità cauzione resa |
| _MMDES$(R) | Descrizione articolo; "" = AUTO |
| _MMCMR(R) | Causale -v._MMCMO-/Contropartita -max 32 x Doc. |
| _MMRMAG(R) | Da Magazzino *d |
| _MMRAMA(R) | A Magazzino *d |
| _MMTIP$(R) | Um; "1"/""=Um1, "2"=Um2, "V"=V.Fisso |
| _MMCOL(R) | Numero colli |
| _MMQTA(R) | Quantità/Peso Lordo |
| _MMGESTLOT$(R) | Lotti- tipo gest.lotto; se = "" azzera dati lotto |
| _MMIDLOT(R,L) | Lotti- Idlotto; -1 autocod.car., -2 autoprel.scar. |
| _MMLNDQTA(R,L) | Lotti- Numero decimali quantità |
| _MMLQTA(R,L) | Lotti- Quantità |
| _MMLCOLLI(R,L) | Lotti- Colli) |
| _MMLQTATGL(R,L,Tg) | Lotti - Quantità per taglia |
| _MMKTA(R) | Koef/Tara su mov.; in PUT se -1=>0, se 0=>anagrafica |
| _MMNCAUZ$(R) | Imballo variabile in emissione |
| _MMTAG(R,16) | Quantità per taglia dalla 1 alla 16 |
| _MMTAGXL(R,16) | Quantità per taglia dalla 17 alla 32 |
| _MMORD$(R) | Stato riga d'ordine E,N,S,B *e |
| _MMCAN(R) | Categ. annullamento; _MMORD$="n"/"N" |
| _MMPRZ(R) | Prezzo |
| _MMSCO$(R) | Sconto |
| _MMINIRARI$(R) | Data inizio Rateo Risconto |
| _MMFINRARI$(R) | Data fine Rateo Risconto |
| _MMSCA$(R) | Data scadenza |
| _MMSOR$(R) | Scadenza ordine origine |
| _MMCONS$(R) | Data prevista consegna - doc.sospesi - solo produz. |
| _MMCNOR$(R) | Data prevista consegna - doc.sospesi - solo produz. |
| _MMPROD$(R) | Data prevista produzione - doc.movimento - solo produz. |
| _MMPROR$(R) | Data prevista produzione - doc.movimento - solo produz. |
| _MMPRO(R) | % provvigione agente |
| _MMPROAGE$(5) | Cod.agente; Tipo F/P; se 1^ elemento="-1" azzera tutto |
| _MMPROCOND(5) | Condizione agente |
| _MMPROTP$(5) | Tipo provvigione; % o T -val.totale- |
| _MMPROFORM$(5) | Formula provvig.; valida solo per 1^ elemento |
| _MMPROSA(5) | Calcolo formula provv. - sola lettura - |
| _MMPROQUOTA(5) | Quota provv. ripartita; tot. quote = 100% |
| _MMPROMOD$(5) | Modal. "P"provv./"R"ripart.; solo per 1^ elem |
| _MMPROBIMP$(5) | Importo calcolo provv.; "1"lordo, "2" netto |

| _MMALI$(R) | Aliquota Iva |
|---|---|
| _MMRIT$(R) | Ritenuta d'Acconto S/N |
| _MMTAGTAB(R) | Tabella Taglie |
| _MMCS2(R) | Cat.Num. su mov.; in PUT se -1=>0, se 0=>anagraf. |
| _MMCS1$(R) | Cat.Alfan. su mov.; in PUT se " "=>"", se ""=>anagraf. |
| _MMULT(R) | Costo ultimo allo scarico *f |
| _MMSTD(R) | Costo standard allo scarico *f |
| _MMPON(R) | Costo medio ponder.allo scar. *f |
| _MMSAZOF(R) | sottoazienda doc.OF collegato, vedi .APRIAZIE |
| _MMSOF(R) | num. serie ordine collegato |
| _MMNOF(R) | num. ordine.forn.collegato *g |
| _MMNCRIGAOF(R) | num. creaz.riga OF colleg. –vedi _MMNCRERIGA(N)- |
| _MMNBL(R) | num. bolla.lav.collegata*g |
| _MMRDC(R) | num. rif.doc.collegato*g |
| _MMSTRIGAPV$(R) | Stato riga preventivo |
| _MMPSAZ(R) | Sottoazienda doc.prev.origine, vedi APRIAZIE |
| _MMPSI$(R) | Rif. sigla preventivo |
| _MMPSE(R) | Rif. serie preventivo |
| _MMPNU(R) | Rif. numero preventivo |
| _MMPDA$(R) | Rif. data preventivo |
| _MMCONS$(R) | Data prevista consegna |
| _MMPROD$(R) | Data prevista di produzione |
| _MMNRECRIGA(R) | Numero di record riga; 6 car *h |
| _MMNCRERIGA(R) | Numero di riga univoco nei documenti)*h |
| **Rintracciabilità (in sola lettura)** | |
| _MMRINTRCRER(R) | riferimento PF della BL; solo produzione |
| _MMRINTRDAT$(R) | data documento della BL; solo produzione |
| _MMRINTRBNU(R) | numero BL; solo produzione |
| _MMRINTRBSBL(R) | sotto BL; solo produzione |
| _MMRINTRBFS(R) | fase BL; solo produzione |
| _MMRINTRBPDR(R) | n.sottobolla del PF che ha generato SML; solo prod. |
| _MMRINTRPFPD(R) | num. riferimento PF che ha generato SML; solo prod. |
| _MMRINTRDTP$(R) | data sottobolla PF che ha generato SML; solo prod. |
| _MMRINTRSIG$(R) | sigla doc. da ricercare; solo DBA+prod.; se ="" è BL |
| _MMRINTRSER(R) | serie documento; solo articolo con DBA+produzione |
| _MMRINTRNUM(R) | num. documento; solo articolo con DBA+produzione |
| _MMRINTRCLI$(R) | cliente/forntore; solo articolo con DBA+produz. |
| _MMRINTRNUMT(R) | progressivo testata valido se _MMRINTROSI$() vuoto Righe riferimento ordine origine del documento da ricercare; Righe solo per articolo con DBA + produzione |
| _MMRINTROSI$(R) | sigla documento ordine origine; art. DBA+prod. |
| _MMRINTROSE(R) | serie ordine origine; solo art. con DBA+produz. |
| _MMRINTRONU(R) | numero ordine origine; solo art. con DBA+produz. |
| _MMRINTRODA$(R) | data ordine origine BL in ricerca; solo art.DBA+prod. |
| **Variabili RIGMM** | |
| _MMR_QTA | Quantità/peso lordo |
| _MMR_IMPON | Imponibile |
| _MMR_PREZZO | Prezzo |
| _MMR_COLLI | Colli |
| _MMR_TARA | Tara |
| _MMR_SCONTO | Valore sconto |
| _MMR_OMAnnn | Merce omagg. tipo nnn=100,110,120,130,200 |
| _MMR_PROVV | Importo provvigione |
| _MMR_PROVSA(15) | Provvigione singolo agent |
| _MMR_PROAGE$(15) | Cod. agente provvigione |
| _MMR_IMPNT | Importo netto di riga |
| _MMR_PSMER(9) | Percentuale sconto sulla riga (valido per gli sconti a cascata, a valore, a percentuale, con sconto >100 e < 109, con sconto >130 e < 139, con sconto >200 e < 209 |
| _MMR_VSMER(9) | Valore dello sconto |
| _MMR_PSMEP | Percentuale sconto pagamento merce |
| _MMR_VSMEP | Valore dello sconto pagamento merce |
| **PIEDE DEL DOCUMENTO** | |
| **Riferimenti Al Trasporto** | |
| _MMSPE$(T) | Codice indir.spedizione |
| _MMANSPE$(T) | Codice da anagraf.indir.spediz. |
| _MMTSR$(T) | Trasporto a mezzo -solo movimento- |
| _MMVET$(T) | Codice vettore |
| _MMCOMM$(T) | Committente |
| _MMCAR$(T) | Caricatore |
| _MMPROP$(T) | Proprietario |
| _MMPOR$(T) | Tipo porto |
| _MMINCOT$(T) | Codice Incoterm |
| _MMDSP$(T) | Ulteriore descriz.porto |
| _MMTPS$(T) | Tipo spese di spedizione |
| _MMVAS(T) | Valore spese di spedizione |
| _MMACO$ | Calcolo automatico colli S/N |
| _MMTCO(T) | Totale colli |
| _MMAPE$ | Calcolo automatico peso S/N |
| _MMTPE(T) | Totale peso |
| _MMASP(T) | Tab.aspet.est.beni -solo movimento- |
| _MMDTINTR$(T) | Data inizio trasporto |
| _MMORINTR$(T) | Ora inizio trasporto |
| **Appalti Pubblici** | |
| _MMCIG$ | Codice CIG |
| _MMCUP | Codice CUP |
| _MMNCTRIMPR$ | Iva MOSS Nota credito riferita ad operazioni trimestri precedenti |
| _MMSPLITPAY$ | ='S' Effettua sempre lo Split Payment. Significa che in creazione del documento il cliente ha le seguenti condizioni ENTE PUBBLICO = S e ENTE ESCLUDI SPLIT= N.<br>='s' Effettua sempre lo Split Payment. Significa che in creazione del documento il cliente ha le seguenti condizioni ENTE PUBBLICO = S e ENTE ESCLUDI SPLIT= S.<br>='N' Non effettua mai lo Split Payment. Significa che in creazione del documento il cliente ha le seguenti condizioni ENTE PUBBLICO = S e ENTE ESCLUDI SPLIT= N.<br>='n' Non effettua mai lo Split Payment. Significa che in creazione del documento il cliente ha le seguenti condizioni ENTE PUBBLICO = S e ENTE ESCLUDI SPLIT= S.<br>='E' Effettua lo Split Payment in automatico e significa che in creazione del documento il cliente ha le seguenti condizioni ENTE ENTE PUBBLICO = S e ENTE ESCLUDI SPLIT= N<br>='e' Non effettua lo Split Payment e significa che in creazione del documento il cliente ha le seguenti condizioni ENTE PUBBLICO = S e ENTE ESCLUDI SPLIT= S<br>=' 'Se si sta scrivendo un nuovo documento, verifica le condizioni del cliente ed applica se è possibile lo split payment. Se si sta revisionando un documento, non esegue nessuna operazione.<br>='L' Valido solo in revisione di un documento esegue la lettura dei dati del cliente e se possibile viene applicato lo split payment. |
| **Pagamenti** | |
| _MMPAG | Codice pagamento |

| | |
|---|---|
| _MMDECOR$ | Data decorrenza pagamento |
| _MMBAPP | Numero banca di appoggio -F5- |
| _MMCODRID | Codice mandato RID SEPA |
| _MMACC | Acconto/pagato |
| _MMABB | Abbuono |
| _MMCTRACC$ | Contropartita Acconto *i |
| _MMCTRABB$ | Contropartita Abbuono *i |
| _MMALISPBAN$ | Aliq. Iva spese banca |
| _MMSPBANCA | Valore spese banca *l |
| _MMSPBOLLI | Valore spese bolli *l |
| **Dati Intracee** | |
| _MMNELINT$ | Escludi elab. Intracee |
| _MMNTR$ | Natura trans. Intracee |
| _MMRGM$ | Condiz. consegna Intracee |
| _MMMTR$ | Modalità trasp. Intracee |
| _MMVAOV | Ammontare valuta *m |
| _MMCAOV | Cambio ammontare in valuta *m |
| _MMSTPIFORN$ | Stato P.Iva Forn. autofattura, doc. -FF/NF- |
| _MMCDPIFORN$ | Codice P.Iva Forn. autofattura, doc. -FF/NF- |
| _MMRIFSERI | Serie doc. fatt. riferimento Intra -NF/NC- |
| _MMRIFNUMI | Numero doc. fatt. riferimento Intra -NF/NC- |
| _MMRIFNUREI$ | Num. Alfan. doc. fatt. riferim Intra -NF/NC- |
| _MMRIFDTI$ | Data doc. fatt. riferimento Intra -NF/NC- |
| _MMPROME$ | Provincia destinazione merce intracee |
| **Enasarco** | |
| _MMCRNAS | Contributo enasarco; se=-2 è ricalcolato |
| _MMCENAS | Importo enasarco con spese banca o bolli |
| **Variabili TOTMM** | |
| _MMTIMPENAS | Imponibile soggetto a enasarco |
| _MMT_CENAS | Contributo enasarco |
| _MMT_PENAS | Percentuale enasarco a carico azienda |
| _MMT_TIMPON | Totale imponibile |
| _MMT_TIMPOS | Totale imposta |
| _MMT_PCALIQ$(6) | Aliquota iva |
| _MMT_PNALIQ(6) | Numero iva; solo aliquota num |
| _MMT_PIMPON(6) | Imponibile; somma contropartite |
| _MMT_PIMPOS(6) | Imposta; somma contropartite |
| _MMT_SPBANCA | Spese banca |
| _MMT_SPBOLLI | Spese bolli |
| _MMT_SPTRASP | Spese trasporto |
| _MMT_SPESE | Spese; articoli tipo S |
| _MMT_CAUZVEN | Cauzioni vendute |
| _MMT_MERCE | Valore merce |
| _MMT_SCONTI | Valore sconti |
| _MMT_OMAnnn | Merce omagg. tipo nnn=100,110,120,130,200 |
| _MMT_PROVVI | Importo provvigione |
| _MMT_PROVSA(15) | Importo provvigione singolo agente |
| _MMT_PROAGE$(15) | Cod. agente provvigione |
| _MMT_COLLI | Colli |
| _MMT_PESO | Peso |
| _MMT_CAUZCON(32) | Cauzioni consegnate |
| _MMT_CAUZVAL | Valore cauzioni; consegnate-rese |
| _MMT_ACCONTO | Acconto |
| _MMT_PSCVAL | Percentuale sconto in valuta |
| _MMT_VSCVAL | Valore sconto in valuta |
| _MMT_PSCMER | Percentuale sconto merce |
| _MMT_VSCMER | Valore sconto merce |
| _MMT_IMPRIT | Imponibile ritenuta d'acconto |
| _MMT_RITAC | Importo ritenuta d'acconto |
| _MMT_IMPCPR | Imponibile cassa professionisti |
| _MMT_CAPRO | Importo cassa professionisti |
| _MMT_TSPPAY | Imposta Split Payment |
| **Videate Utente** | |
| _MMVDT$(T,26) | 26 campi videata di testata |
| _MMVDR$(R,26) | 26 campi videata di riga; max 627 articoli) (la GET converte da num->str$, la PUT viceversa |
| **VARIABILI CHE POSSONO ESSERE UTILIZZATE PER LA MODULISTICA DOCUMENTI** | |
| _MMSXN(1-20) | Variabile numerica di testata/piede |
| _MMSXA$(1-20) | Variabile Alfanumeri di testata/piede |
| _MMSWN(R,1-10) | Variabile numerica di riga |
| _MMSWA$(R,1-10) | Variabile Alfanumerica di riga |
| _MMSWF(R,1-20) | Variabile numerica di riga |
| _MMSWS$(R,1-20) | Variabile Alfanumerica di riga |

In Sprix/Collage sono array numerici e stringa stampati nel MODULO del documento, se appositamente predisposto in modulistica documenti con dei corrispondenti codici di stampa (scelti dalla voce COLLAGE in elenco) di max 14 caratteri (se numerici) o 80 caratteri (se stringa) e relativi a:
- Testata/Piede Modulo (corrispondono ai monodimensionali):
(numerici XN01, XN01,..., XN20) e (stringa XA01, XA02,..., XA20);
- Corpo Modulo (corrispondono ai bidimensionali, _NMM è numero di riga):
(numerici WN01, WN02,..., WN10) e (stringa WA01, WA02,..., WA10);

Esempio, stabiliti nel modulo: XN01 di testata e WA01 di corpo
Sprix: _MMMODSTP$="2": _MMSXN(1)=33: _MMSWA$(1,1)="new": PUTMM 4: END
Collage: ON_PUTMM: _MMSXN(1)=33: _MMSWA$(1,1)="new": END

Gli array possono essere azzerati da operazioni GET o se inseriti dopo articoli descrittivi; fuori dal modulo Magazzino utilizzare _SST$ o _TSST$.

| | | |
|---|---|---|
| _LAIDL() | Id lotto caricati in riga documento | Collage: ON_LISTALOTTI Lotti |
| _LARIG | Num. riga che si sta processando | Collage: ON_LISTALOTTI Lotti |
| _LANUM | Num. lotti gestiti nella riga | Collage: ON_LISTALOTTI Lotti |

(NB: variabili _LA? si azzerano con AZZVARSYS 15 e non è utilizzabile alcuna interazione a video; il 20^ elemento del dizionario lamov() riporta se il lotto ha subito carichi multipli.

Esempio:

lamov(20)(1)= se il lotto 1 ha carichi multipli da più doc.).

```
Es. FOR I=1 TO _LANUM
I$=STR$(_LAIDL(I),0,0): R$=STR$(_LARIG,0,0): N$=STR$(_LANUM,0,0)
NEXT
```

Per trasformare un documento occorre leggerlo, modificarne gli estremi e riscriverlo ex-novo con concomitante cancellazione dell'originario documento. L'operazione è corretta se le variabili sono impostate con gli accorgimenti richiesti dai documenti e in base alle condizioni di trasformazione (operazioni invece automatiche nello standard), che vengono controllate corrette nella riorganizzazione archivi con messaggi di tipo "Doc.di scarico errato, corretto"); i disallineamenti di archivi sono provocati da grossolane trasformazioni di variabili lette, ma non trattate in scrittura e quindi non adatte a nuovi documenti. Esempio, se i riferimenti del documento non vengono aggiornati, si riferirebbero al documento originario (da cancellare); Sprix non aggiorna i riferimenti, quindi occorre verificare il riferimento per ciascuna riga del documento trasformato: gli array della testata sono _MM?SI$, _MM?SE_MM?NU, _MM?DA$ ed il collegamento di questi riferimenti alle righe è tramite l'array _MMNTE cioè il riferimento della testata che indica il documento di origine di ogni riga. I riferimenti al documento di origine _MM?SI$ sono letti solo se nell'array _MMNTE è indicato l'indice corrispondente ai riferimenti alle testate in modo progressivo crescente e continuo, dato che c'è un collegamento diretto fra le righe di ciascun documento originario e i ri-ferimenti (riportati in testata) nel documento trasformato. Nella lettura e trasformazione di più documenti, le variabili non subiscono lo stesso trattamento: per alcune possono continuare a valere i valori letti, ma per altre possono avvenire azzeramenti di default prima della scrittura (es. _MMVAS() se azzerata, causa l'assenza delle spese di trasporto in primanota), pertanto è utile separare la lettura dalla scrittura salvando i dati su variabili d'appoggio. Non è consentita la fusione di più documenti sospesi in un unico dello stesso tipo (è rilasciato il messaggio "per i docum. prev/ord/matr non è ammessa più di una testata"): occorre trasformarli in uno di movimento (azzerando _MMMSI$(I) per _NTMM righe per impostare correttamente il documento di scarico (non trovato con il documento origine e con il seguente messaggio "Numero di riga univoco utilizzato per la riga n (riga n-1)").

*Note:*

***a** – _MMVAL, _MMEVE, _MMDIE, _MMEVC, _MMDIC solo GETATTMM con valuta documento <> da quella di gestione contabile: se il documento è emesso nella stessa valuta di gestione dell'azienda il Cambio = 1 e il Flag = 0. Dato un importo in valuta di partenza il Flag stabilisce come operare per calcolare l'importo nella valuta di arrivo: se Flag= 0 moltiplicare oppure se = 1 dividere. Es. triangolazione con area Euro (100$->Eu-Lir) dati i cambi $ : Eu = 1,1565:1 e Eu : Lir = 1:1936,27 : (se _MMDIE=1 e _MMEVE=1,1565 -> 100:1,1565 = 86,47 Eu; (se _MMDIE=0 e _MMEVE=0,86467 -> 100*0,86467 = 86,47 Eu -cambio inverso-; Eu -> Lir (_MMDIC=0 e _MMEVC=1936,27) 87,47 Eu*1936,27 = 167,429 Lir

***b** – Caratteri alfanumerici di numerazione e Riferimenti Documento

In primanota e magazzino è possibile utilizzare i caratteri alfanumerici nella numerazione dei documenti che utilizzano il registro acquisti. Sia in prima nota che in magazzino, il dato in input del numero di documento può essere numerico o alfanumerico, ed è gestito a scorrimento: se alfanumerico, in uscita dal campo questo viene elaborato in numerico (c'e una estrapolazione dei soli numeri a partire da destra, per la formazione della chiave numerica del documento), rientrando nel campo viene ripristinata la visualizzazione dell'alfanumerico (in magazzino, il dato è gestito per i nuovi documenti FF e NF ed è gestito anche nel campo "Numero" dei Riferimenti Esterni [F4] e non è svincolato dal numero documento; per i documenti vecchi o emessi senza questa modalità, la gestione rimane inalterata). In magazzino l'input alfanumerico è con _MMNUMRE$, in prima nota è con la variabile _PNNUMRE$; non sono obbligatorie, se impostate devono contenere almeno un carattere numerico; l'input da gestionale imposta il numerico e in Sprix, ciò avviene quando la variabile numerica (cioè _PNNDO o _MMNUM ) è = 0; diversamente, se quest'ultime sono <> 0 avviene un controllo di congruità fra numerico e alfanumerico in fase di PUT. Se il dato alfanumerico corrisponde a un numero di max 6 caratteri numerici,il numero è riportato nella variabile numeriche e la variabile alfanumerica è azzerata.

RIFERIMENTI DOCUMENTI E RIFERIMENTI DOCUMENTI ESTERNI

Nella trasformazione di più documenti sospesi in uno di ordine superiore occorre azzerare la variabile _MMMSI$(T)="" per _NTMM(righe). Sulle variabili dei riferimenti c'è un controllo nell'inserimento del documento magazzino: se il tipo documento di scarico uguale al tipo documento, vengono forzati la data/numero/serie documento di scarico e riferimenti esterni documento di scarico; se il tipo documento di scarico di tipo movimento magazzino è diverso dal tipo documento, viene controllata la data di scarico che sia entro i limiti di gestione dell'anno fiscale. Notare: il semplice azzeramento delle variabili _MMO? e _MMM? rende comunque nuova la composizione del documento, seppur inizializzata da GET.

***c** – _MMTPR$

Se = "" cancella la riga; 'r' è in trasform. doc. letto con GETATTMM

***d** – _MMRMAG(R), _MMRAMA(R)

Se tutti i 'Magaz' di riga sono uguali, sono valorizzati di default dal 'Magaz' di testata; se fra loro differenti sono invece prioritari rispetto a _MMMAG (valorizzato con ultimo _MMRMAG() di riga). Gli 'A Magaz' di riga non seguono questa logica (perchè può essere = 0). Anche una sola valorizzazione di riga <> da 0 obbliga l'utilizzo e la valorizzazione di tutte le _MMRAMA(R).

***e** – _MMORD$(R)

In doc. fornitori che non derivano da trasformazione, _MMORD$()="" e articoli considerati Evasi; per questi doc. _MMORD$ è gestita solo in trasformazione, in cui definire quali/quanti art. devono essere evasi.

***f** – _MMULT(R), _MMSTD(R), _MMPON(R)

Se in fase di scrittura sono impostate a -1 sono assunti come costi (ultimo, stand. e medio ponder.) quelli di anagrafica al momento della scrittura, altrimenti viene assunto il valore che viene impostato.

***g** – _MMNOF(R), _MMNBL(R), _MMRDC(R)

queste variabili non vanno modificate; se il valore di _MMNOF o _MMNBL è <> da zero, nessuna variabile di quella riga può essere modificata

***h** – _MMNRECRIGA(R), _MMNCRERIGA(R)

_MMNRECRIGA(R) serve a mantenere DBA/DBVO nelle righe di doc. sospesi e di movimento. È letta in GETMM e utilizzabile in PUTMM, che la valorizza con i relativi num. record; non gestita in imp/exp ASCII-CSV (come anche DBA/DBVO), nè in Dbmirror/Ciclotabellare (nel ciclotab. si legge con la ..nri() dei documenti sospesi). DBVO è gestita solo su articoli con DBP. La variabile può assumere valori >=< 0 :

**_MMNRECRIGA>0** (default): mantiene DBA/DBVO gestite su doc.precedenti; se modificata è perso il collegamento alla stessa riga e non può ri salire alla DBA/DBVO da mantenere nel documento registrato con PUTMM. Effetti in base all'operatività:

- GETMM e PUTMM dello stesso documento sospeso: se lasciato invariato il valore ogni riga articolo che gestisce DBA/DBVO manterrà quella del documento originario.
- Impostazione variabili _MM? e PUTMM di documento sospeso: se forzata la variabile a > 0 è mantenuta quella originaria dell'eventuale documento in archivio relativo alle variabili _MM?; se non esiste NON viene generata.
- GETMM di documento sospeso e PUTMM di documento di movimento: se lasciato invariato il valore e se il documento di movimento gestisce la DBA, quest'ultimo genera i documenti di scarico e carico relativi alla DBA presenti nel documento sospeso.
- Impostazione variabili _MM? e PUTMM di documento di movimento: se impostate le variabili relative al documento origine di tipo sospeso e se il documento di movimento gestisce la DBA, quest'ultimo genera i doc. di scarico e carico relativi alla DBA presenti nel doc. sospeso

**_MMNRECRIGA=0** (default release precedenti): non sono gestite DBA/DBVO sui documenti, le righe perdono le DBA/DBVO. Effetti in base all'opera tività:

- GETMM e PUTMM dello stesso documento sospeso: se forzata la variabile a 0, ogni riga articolo che gestisce la DBA/DBVO la perde, cancellando quella originaria del documento.
- Impostazione variabili _MM? e PUTMM di documento sospeso: se forzata la variabile a 0, non viene gestita la DBV/DBVO.
- GETMM di documento sospeso e PUTMM di documento di movimento: se impostato il valore della nuova variabile a 0, se il documento di movimento gestisce la DBA, NON genera i documenti di scarico e carico relativi alla DBA.
- Impostazione variabili _MM? e PUTMM di documento di movimento: se impostate le variabili relative al documento origine di tipo sospeso e se il documento di movimento gestisce la DBA quest'ultimo NON genera i documenti di scarico e carico relativi alla DBA. Questo vale anche se il documento di movimento gestisce la DBA e NON presenta le variabili del documento origine.

**_MMNRECRIGA<0** (-1): viene letta da anagrafica articoli la relativa DBA. Effetti in base all'operatività:

- GETMM e PUTMM dello stesso documento sospeso: se forzata la variabile a -1, ogni riga articolo che gestisce la DBA/DBVO caricherà quella di anagrafica e cancellerà quella originaria del documento.
- Impostazione variabili _MM? e PUTMM di documento sospeso: se forzata la variabile a -1, viene caricata la DBV/DBVO dell'anagrafica.
- GETMM di documento sospeso e PUTMM di documento di movimento: se impostato il valore della nuova variabile a -1, se il documento di movimento gestisce la DBA, genera i documenti di scarico e carico relativi alla DBA definita in anagrafica e perde quella del doc.sospeso.- Impostazione variabili _MM? e PUTMM di documento di movimento: se impostate le variabili relative al documento origine di tipo sospeso e se il documento di movimento gestisce la DBA, quest'ultimo genera i documenti di scarico e carico relativi alla DBA definita in anagrafica, perdendo quella del documento sospeso. Questo vale anche se il documento di movimento gestisce la DBA e NON presenta le variabili del documento origine.

_MMNCRERIGA(R) è il numero di riga univoco nei documenti (v. diziona rio mmrig() e ocrig()). È un numero assegnato in fase di creazione riga (di tipo, articolo, testo e descrizione). Se la riga viene cancellata il numero viene perso. Se nella riga viene modificato il codice ar ticolo (o il tipo di riga) la procedura assegna un nuovo num. univoco.

Le logiche di gestione del dato quando non è modificato (diversamente la procedura controlla sia univoco e lo assegna alla riga se non diversamente specificato) sono le seguenti:

Con la generazione di nuovo documento non derivante da sottodocumenti (es. BC 1) la procedura assegna il numero univoco di riga progressivo a partire da 1 (1,2,3..), anche se venisse imposta la variabile.

Es.:

-> OC di 3 righe (ID=_MMNCRERIGA) -> ID riga 1, ID riga 2, ID riga 3

La variazione codice articolo/tipo alla riga 2 provoca rinumerazione:

-> ID riga 1, ID riga 4, ID riga 3 (la riga 2 è rinumerata)

Se il documento è nuovo, ma deriva da sottodocumenti (es. OC1 e OC2 in BC1) e con aggiunta di nuove righe, la procedura assegna il numero solo alle righe che hanno la _MMNCRERIGA = 0, lasciando quello origine alle restanti.

Con la trasformazione di un documento sospeso in un altro sospeso, la riga articolo mantiene lo stesso numero univoco;

Con la trasformazione/fusione di più documenti sospesi in altro sospeso, la procedura assegna alle righe articolo un numero univoco progressivo che parte da 1.

Esempio fusione 2 documenti (OC 1 + OC2) in OC 3

| OC1 + | OC2 | -> OC3 (N.B. le righe vengono rinumerate) |
|---|---|---|
| ID riga 3 | ID riga 1 | ID riga 1 |
| ID riga 6 | ID riga 4 | ID riga 2 |
| ID riga 7 | ID riga 6 | ID riga 3 |
| ID riga 4 | | |
| ID riga 5 | | |
| ID riga 6 | | |

Con la trasformazione di uno/più documenti sospesi in uno di movimento, il numero di riga univoco delle righe del movimento è mantenuto uguale a quello di riga del documento sospeso origine.

Esempio

| OC 1 | -> BC1 | |
|---|---|---|
| ID riga 3 | ID riga 3 | questo vale anche per le traformazioni |
| ID riga 4 | ID riga 4 | multiple (es.OC1+OC2 in BC1) |
| ID riga 7 | ID riga 7 | |

Con evasione parziale di OC (gestione da Emissione-Revis. documenti)

ID riga 3: 10 pezzi art A e ID riga 5: 10 pezzi art B

l'evasione di 2 pezzi della 1^ riga in BC ha ID 3 (che resta anche per gli 8 pezzi dell'art A sull'OC)

Con documenti di movimento derivati da più documenti: il numero di riga è univoco all'interno della sottotestata dei documenti:

BC 1 Articolo: A numero di riga univoco: 1

Articolo: B numero di riga univoco: 2

BC 2 Articolo: C numero di riga univoco: 1

Articolo: D numero di riga univoco: 2

FT1 derivata da BC1 + BC2

Articolo: A numero di riga univoco: 1 rif BC1

Articolo: B numero di riga univoco: 2 rif BC1

Articolo: C numero di riga univoco: 1 rif BC2

Articolo: D numero di riga univoco: 2 rif BC2

***i** – _MMCTRACC$, _MMCTRABB$

Per definire controparite diverse da quelle dei conti automatici per ogni documento; se impostate a "-1" vengono prelevate le contropartite relative al codice pagamento del documento (se èimpostato,altrimenti,se impostate a "-1" ma manca il codice del pagamento sono prese quelle dei conti automatici).

***l** – _MMSPBANCA, _MMSPBOLLI

La PUTMM di documenti nuovi calcola le spese banca e bolli in base a quanto stabilito dal codice pagamento impostato e valori di tabella. Nella revisione l'aliquota e le spese sono modificabili; dalla versione 50.1 nella variazione di documento già presente, immesso un valore questo viene aggiornato, ma occorre aggiornare la primanota:

se _MMSPBANCA viene impostato -1 allora sia l'aliquota che il valore delle spese banca sono impostate con i valori presenti in tabella; se a variabile _MMSPBOLLI viene impostato -1 allora il valore delle spese bolli viene impostato con il valore presente in tabella. Se le variabili sono impostate con valore -2 la relativa spesa (banca/bollo) è azzerata (sia in caso di documento nuovo che trasformato)

***m** – _MMVAOV, _MMCAOV

Se _MMCAOV = 0 e _MMVAOV<>0 È prelevato il cambio relativo alla data ingresso terminale

Esempio: Registra un OC e ne legge alcuni totali. Registra un secondo OC con DBA e aggiorna il primo cancellando+aggiungendo una riga articolo. Evade i due OC in BC aggiungendo la provvigione. Visualizza riferimenti delle righe della BC e cancella gli OC; infine procede alla fatturazione con cancellazione di BC e visualizzazione dei riferi- menti del movimento.

```
_MMSIG$="OC":_MMSER=1:_MMNUM=0:_MMDAT$=sxdat:_MMCLI$="201.00001":_MMMAG=1
_MMTPR$(1)="R":_MMART$(1)="04M-MI":_MMQTA(1)=1:_MMPRZ(1)=100:_MMALI$(1)="20"
_MMTPR$(2)="R":_MMART$(2)="04M-SI":_MMQTA(2)=2:_MMPRZ(2)=200:_MMALI$(2)="20"
_MMRICSTP=1: PUTMM 2:IF _ERRMM<>0 THEN GOTO ERRORE
RIGMM 1,2,2:TOTMM
VIMSG 1,"Prez."+STR$(_MMR_PREZZO,2,0)+"|Imponib."+STR$(_MMR_IMPON,2,0)
VIMSG 1,"Totale Imponibile "+STR$(_MMT_TIMPON,2,0)
OSIG1$=_MMSIG$ : OSER1$=STR$(_MMSER,0,0) : ONUM1$=STR$(_MMNUM,0,0)

AZZVARSYS 4
_MMSIG$="OC":_MMSER=1:_MMNUM=0:_MMDAT$=sxdat:_MMCLI$="201.00001":_MMMAG=1
_MMTPR$(1)="R":_MMART$(1)="PCDESK":_MMQTA(1)=1:_MMPRZ(1)=300:_MMALI$(1)="20"
_MMNRECRIGA(1)=-1:PUTMM 2: IF _ERRMM<>0 THEN GOTO ERRORE
OSIG2$=_MMSIG$ : OSER2$=STR$(_MMSER,0,0) : ONUM2$=STR$(_MMNUM,0,0)

GETMM_EXT OSIG1$+OSER1$+"/"+ONUM1$,"201.00001"
IF _ERRMM<>0 THEN GOTO ERRORE
_MMTPR$(1)=""
_MMTPR$(3)="D":_MMTSD$(3)=".":_MMDEE$(3)="Riga descrittiva"
SETPARSOSTMATR "S":PUTMM 0: IF _ERRMM<>0 THEN GOTO ERRORE

AZZVARSYS 4
GETMM OSIG1$+ONUM1$+"+"+OSIG2$+ONUM2$ : IF _ERRMM<>0 THEN GOTO ERRORE
_MMSIG$="BC ": _MMSER=1 : _MMNUM=0 : _MMAGE$="401.00001"
FOR I=1 TO 3 : _MMORD$(I)="E": _MMPRO(I)=5 : NEXT
PUTMM 2 : IF _ERRMM<>0 THEN GOTO ERRORE
BSIG$=_MMSIG$ : BSER$=STR$(_MMSER,0,0) : BNUM$=STR$(_MMNUM,0,0)

AZZVARSYS 4
GETMM BSIG$+BNUM$
 VIMSG 1, "visualizzo i Riferimenti delle righe della bolla"
 FOR I=1 TO _NMM
 VIMSG 1,_MMART$(I)+"->"+_MMOSI$(_MMNTE(I))+STR$(_MMONU(_MMNTE(I)),0,0)
 NEXT

AZZVARSYS 4
DELMM_EXT OSIG1$+OSER1$+"/"+ONUM1$+"+"+OSIG2$+OSER2$+"/"+ONUM2$,"201.1"
IF _ERRMM<>0 THEN GOTO ERRORE

AZZVARSYS 4
GETMM BSIG$+BSER$+"/"+BNUM$ : IF _ERRMM<>0 THEN GOTO ERRORE
_MMSIG$="FT": _MMSER=1 : _MMNUM=0
PUTMM 1 : IF _ERRMM<>0 THEN GOTO ERRORE

AZZVARSYS 4
GETMM BSIG$+BNUM$
 VIMSG 1, "visualizzo i Riferimenti delle righe della fattura"
 FOR I=1 TO _NMM
 VIMSG 1,_MMART$(I)+"->"+_MMMSI$(_MMNTE(I))+STR$(_MMMNU(_MMNTE(I)),0,0)
 NEXT

DELMM BSIG$+BSER$+"/"+BNUM$ : IF _ERRMM<>0 THEN GOTO ERRORE
ERRORE: IF _ERRMM<>0 THEN VIMSG 4, _ERRMM$
```

# _MOD

variabili di Import Moduli

| | |
|---|---|
| _MODTOTRIG | Totale righe file modulo di import |
| _MODNUMRIG | Numero della riga letta con la MODGET, dopo la MODOPEN |
| _MODFINEP | Flag di fine pagina, 1 = la riga letta è l'ultima della pagina |
| _MODFINEC | Flag di fine corpo pagina, se gestito, 1 = riga letta è l'ultima riga della pagina |
| _MODNUMRIGP | Numero della riga di pagina letta con la MODGET |
| _MODNUMRIGC | Numero della riga del corpo pagina letta con la MODGET |
| _MODNUMRIGLC | Numero della riga logica del corpo pagina letta con la MODGET |
| _MODSCRIVI | Variabile che indica, quando impostato a 1, la necessità di scrittura dei dati, dovuta a fine pagina o fine corpo, oppure, se definito, per rottura dei valori dei campi selezionati |
| _ERRMOD | Diverso da zero indica errore bloccante di elaborazione modulo |
| _ERRMOD$ | Descrizione dell'errore _ERRMOD; il controllo è alla MODCLOSE |
| _MSGMOD$() | Array di eventuali messaggi di avviso raccolti durante l'elaborazione, visualizzabili mediante la MODVIMSG |

# _MSC

variabili di selezione campi Import Moduli SC (Saldi Contabili)

| | |
|---|---|
| _MSCCOD$ | Codice conto |
| _MSCDESC$ | Descrizione conto |
| _MSCINIZ | Saldo iniziale |
| _MSCSEZI$ | Sezione saldo iniziale (_MSCINIZ) |
| _MSCDARE | Saldo dare |
| _MSCAVER | Saldo avere |
| _MSCSALD | Saldo dare/avere |
| _MSCSEZS$ | Sezione per Saldo dare/avere (_MSCSALD) |

# _NFIN

In Collage è il numero della finestra attiva in quel momento, se >0 è relativa al programma collage, se <0 è relativa al gestionale. In Sprix il numero di finestra è gestito con il codice del programma Sprix.

# _NRDATI

In Sprix/Collage riporta il numero di record del file ad indici a seguito di operazioni GETKEY, NEXTKEY, PREVKEY, VARKEY, INSKEY, DELKEY.

# _OB

_OB? variabili di struttura archivi Testata Bolla Lavorazione (GETBL-PUTBL-DELBL - GETATTBL)

| | |
|---|---|
| _OBNUM | (numero BL - valorizzata in PUT -) |
| _OBNSO | (numero sottoBL - valorizzata in PUT -) |
| _OBNSP | (num. sottoBL padre, come _OBNSO, -1 per le BL x.0000) |
| _OBDAT$ | (data documento BL) |
| _OBMAG | (magazzino PF - prodotto finito -) |
| _OBMAT | (magazzino MP - materia prima -) |
| _OBSCA$ | (data scadenza BL) |
| _OBDST$ | (nota BL) |
| _OBPIA$ | (piano di lavoro S/N) |
| _OBCTO$ | (codice conto) |
| _OBTIP$ | (tipo modulo di stampa) |
| _OBSTA$ | (stampa della BL, se="S" non esegue) |
| _OBVAL | (totale valore scarichi) |
| _OBINF$ | (informazioni del PF) |
| _OBPRZ$ | (fissa prezzi S/N) |
| _OBASS | (fase assegnata) |
| _OBDBP$ | (codice DBP -se unico-) |
| _OBOPE$ | (tipo operazione impostata -solo GETATTBL-) |
| _OBSML$ | (Cat.SML da sviluppare [F7], valori A-Z) |

_OL? variabili di struttura archivi Righe Bolla Lavorazione (GET-PUT-DEL)

| | |
|---|---|
| _OLTPR$() | (tipo riga)*a |
| _OLLAV() | (numero sottoBL -stottobolla -)*b |
| _OLART$() | (codice PF) |
| _OLPFP$() | (codice PF Padre) |
| _OLRIF() | (numero di riferimento)*c |
| _OLRII() | (numero riferimento impegni) |
| _OLCST() | (totale valore scarichi) |
| _OLCLI$() | (cod. cliente) |
| _OLSPE$() | (cod. indirizzo di spedizione) |
| _OLDBP$() | (cod. DBP) |
| _OLSDO() | (serie OC) |
| _OLORD() | (numero OC) |
| _OLDAT$() | (data documento OC) |
| _OLSCO$() | (data scadenza OC) |
| _OLSCA$() | (data scadenza riga) |
| _OLNTG() | (tipo riga quantità)*d |
| _OLDEC() | (num. decimali) |
| _OLCOL() | (colli) |
| _OLQTF() | (quantità) |
| _OLTAG() | (quantità per taglia) |
| _OLTAGXL() | (quantità per taglia 17-32) |
| _OLCOE() | (coeff. conversione um1/um2) |
| _OLFSC() | (ultima fase scaricata) |
| _OLFSI() | (ultima fase impegnata) |
| _OLFSD() | (ultima fase depositata) |
| _OLFST() | (ultima fase stampata) |
| _OLCBL() | (num. sottoBL o sottoBL colleg lavorante) |
| _OLCFS() | (precedente fase scaricata nella BL lavorante) |
| _OLMAG() | (magazzino caricato) |
| _OLNRX() | (numero record) |
| _OLVAL() | (valore corrente scarico) |
| _OLUNI() | (numero univoco di riga) |
| _OLUNP() | (numero univoco di riga PF) |
| _OLFAS() | (num. fase impostata su riga PF; solo GETATTBL) |
| _OLLOT() | (identificativo del lotto) |
| _OLARE$() | (Codice Area) |
| _OLCOM$() | (Codice Commessa) |
| _OLTDO$() | (Tipo Documento collegato) |
| _OLSTR$() | (Stato di riga) |
| _OLNOT$() | (Nota di riga) |

| _OLSMLPF$() | (Categoria sviluppo SML per PF) |
|---|---|

*Note:*

*a - "R"=normale; "r"=normale bloccata -BL lavorante-; "P"=PF sospeso -solo con sottoBL-; "S"=SML semilavorato sospeso;

*b - =0 SML sospeso; >0 in lavorazione (32000 per le BL x.0000); <0 già caricato (32000 per le BL x.0000);

*c - 1-16000000 collegata a OC; >16000000 produzione per il magazzino;

*d - (0=no taglie, 1-99 taglie);

Esempio: scrive una BL, la legge e la spezza, aggiunge una riga da collegare a un ordine cliente

```
_OBNUM=25:_OBNSP=-1:_OBMAG=2:_OBMAT=1:_OBDAT$="20210101"
_OLLAV(1)=32000:_OLRIF(1)=16000001:_OLRII(1)=1
_OLCLI$(1)="201.29":_OLTPR$(1)="R":_OLART$(1)="ARTDBPTAG":
_OLNTG(1)=2 : Q=7 : FOR T=1 TO 4 : _OLTAG(1,T)=Q : NEXT
_OLLAV(2)=32000:_OLRIF(2)=16000002:_OLRII(2)=2
_OLCLI$(2)="201.28":_OLTPR$(2)="R":_OLART$(2)="MON-CTX14":_OLQTF(2)=3
PUTBL : IF _ERRBL<>0 THEN VIMSG 5, _ERRBL$
VIMSG 1, "Ha creato la Bolla di Lavoro 25.0000"
AZZVARSYS 6
GETBL 25,0 : IF _ERRBL<>0 THEN VIMSG 5, _ERRBL$
FOR I=1 TO _NBL
'REM if _OLLAV(I)=0 then salta (...con SML sospesi, considerati in stampa)
VIMSG 1,"Riga "+STR$(I,0,0)+" Art: "+_OLART$(I)+" Qtà:"+STR$(_OLQTF(I)
NEXT
SPRIGBL 25,0,"ARTDBPTAG",olrii(25,-1,1),2,2,2,2
IF _ERRSPRIGBL<>0 THEN VIMSG 1, _ERRSPRIGBL$
VIMSG 1, "Ha spezzato la riga del PF a taglie della BL"
AZZVARSYS 6
GETBL 25,0 : IF _ERRBL<>0 THEN VIMSG 5, _ERRBL$
NR=_NBL+1
_OLLAV(NR)=32000:_OLRIF(NR)=(16000000+NR):_OLRII(NR)=NR
_OLTPR$(NR)="R":_OLART$(NR)="ARTDBP":_OLQTF(NR)=15
_OLSDO(NR)=1:_OLORD(NR)=290:_OLDAT$(NR)="20210101"
_OLSCO$(NR)="20210424":_OLSCA$(NR)="20210424"
PUTBL : IF _ERRBL<>0 THEN VIMSG 5, _ERRBL$
VIMSG 1, "Ha aggiunto una riga di PF da collegare a un OC"
IF ocnum()=290 THEN N=ocnri() '(NB: da ciclotabellare su tabella OC)
COLOCBL 25,NR,N : IF _ERRCOLOCBL<>0 THEN VIMSG 4, _ERRCOLOCBL$
VIMSG 1, "Ha collegato la riga PF all'OC | ora cancella una BL"
DELBL 22,0 : IF _ERRBL<>0 THEN VIMSG 5, _ERRBL$
```

# _OL

_OB? variabili di struttura archivi Testata Bolla Lavorazione (GET-PUT-DEL)

| _OBNUM | numero BL - valorizzata in PUT |
|---|---|
| _OBNSO | numero sottoBL - valorizzata in PUT |
| _OBNSP | num. sottoBL padre, come _OBNSO, -1 per le BL x.0000 |
| _OBDAT$ | data documento BL |
| _OBMAG | magazzino PF - prodotto finito |
| _OBMAT | magazzino MP - materia prima |
| _OBSCA$ | data scadenza BL |
| _OBDST$ | nota BL |
| _OBPIA$ | piano di lavoro S/N |
| _OBCTO$ | codice conto) |
| _OBTIP$ | tipo modulo di stampa |
| _OBSTA$ | stampa della BL, se="S" non esegue |
| _OBVAL | totale valore scarichi |
| _OBINF$ | informazioni del PF |
| _OBPRZ$ | fissa prezzi S/N |
| _OBASS | fase assegnata |
| _OBDBP$ | codice DBP -se unico- |
| _OBOPE$ | tipo operazione impostata -solo GETATTBL- |
| _OBSML$ | Cat.SML da sviluppare [F7], valori A-Z |

_OL? variabili di struttura archivi Righe Bolla Lavorazione (GET-PUT-DEL)

| _OLTPR$() | tipo riga *a |
|---|---|
| _OLLAV() | numero sottoBL -stottobolla - *b |
| _OLART$() | codice PF |
| _OLPFP$() | codice PF Padre |
| _OLRIF() | numero di riferimento *c |
| _OLRII() | numero riferimento impegni |
| _OLCST() | totale valore scarichi |
| _OLCLI$() | cod. cliente |
| _OLSPE$() | cod. indirizzo di spedizione |
| _OLDBP$() | cod. DBP |
| _OLSDO() | serie OC |

| | |
|---|---|
| _OLORD() | numero OC |
| _OLDAT$() | data documento OC |
| _OLSCO$() | data scadenza OC |
| _OLSCA$() | data scadenza riga |
| _OLNTG() | tipo riga quantità *d |
| _OLDEC() | num. Decimali |
| _OLCOL() | Colli |
| _OLQTF() | Quantità |
| _OLTAG() | quantità per taglia |
| _OLTAGXL() | quantità per taglia 17-32 |
| _OLCOE() | coeff. conversione um1/um2 |
| _OLFSC() | ultima fase scaricata |
| _OLFSI() | ultima fase impegnata |
| _OLFSD() | ultima fase depositata |
| _OLFST() | ultima fase stampata |
| _OLCBL() | num. sottoBL o sottoBL colleg lavorante |
| _OLCFS() | precedente fase scaricata nella BL lavorante |
| _OLMAG() | magazzino caricato |
| _OLNRX() | numero record |
| _OLVAL() | valore corrente scarico |
| _OLUNI() | numero univoco di riga. Se gestione collegamento PF = "S" |
| _OLUNP() | numero univoco di riga PF. Riferito al PF padre valido solo per SML, se gestione collegamento PF = "S" |
| _OLFAS() | num. fase impostata su riga PF; solo GETATTBL |
| _OLLOT() | identificativo del lotto |
| _OLARE$() | Codice Area |
| _OLCOM$() | Codice Commessa |
| _OLTDO$() | Tipo Documento collegato |
| _OLSTR$() | Stato di riga |
| _OLNOT$() | Nota di riga |
| _OLSMLPF$() | Categoria sviluppo SML per PF |

*Note:*

*a - "R"=normale; "r"=normale bloccata -BL lavorante-; "P"=PF sospeso -solo con sottoBL-; "S"=SML semilavorato sospeso;

*b - =0 SML sospeso; >0 in lavorazione (32000 per le BL x.0000); <0 già caricato (32000 per le BL x.0000);

*c - 1-16000000 collegata a OC; >16000000 produzione per il magazzino;

*d - (0=no taglie, 1-99 taglie);

Esempio: scrive una BL, la legge e la spezza, aggiunge una riga da collegare a un ordine cliente

```
_OBNUM=25:_OBNSP=-1:_OBMAG=2:_OBMAT=1:_OBDAT$="20210101"
_OLLAV(1)=32000:_OLRIF(1)=16000001:_OLRII(1)=1
_OLCLI$(1)="201.29":_OLTPR$(1)="R":_OLART$(1)="ARTDBPTAG":
_OLNTG(1)=2 : Q=7 : FOR T=1 TO 4 : _OLTAG(1,T)=Q : NEXT
_OLLAV(2)=32000:_OLRIF(2)=16000002:_OLRII(2)=2
_OLCLI$(2)="201.28":_OLTPR$(2)="R":_OLART$(2)="MON-CTX14":_OLQTF(2)=3
PUTBL : IF _ERRBL<>0 THEN VIMSG 5, _ERRBL$
VIMSG 1, "Ha creato la Bolla di Lavoro 25.0000"
AZZVARSYS 6
GETBL 25,0 : IF _ERRBL<>0 THEN VIMSG 5, _ERRBL$
FOR I=1 TO _NBL
'REM if _OLLAV(I)=0 then salta (...con SML sospesi, considerati in stampa)
VIMSG 1,"Riga "+STR$(I,0,0)+" Art: "+_OLART$(I)+" Qtà:"+STR$(_OLQTF(I)
NEXT
SPRIGBL 25,0,"ARTDBPTAG",olrii(25,-1,1),2,2,2,2
IF _ERRSPRIGBL<>0 THEN VIMSG 1, _ERRSPRIGBL$
VIMSG 1, "Ha spezzato la riga del PF a taglie della BL"
AZZVARSYS 6
GETBL 25,0 : IF _ERRBL<>0 THEN VIMSG 5, _ERRBL$
NR=_NBL+1
_OLLAV(NR)=32000:_OLRIF(NR)=(16000000+NR):_OLRII(NR)=NR
_OLTPR$(NR)="R":_OLART$(NR)="ARTDBP":_OLQTF(NR)=15
_OLSDO(NR)=1:_OLORD(NR)=290:_OLDAT$(NR)="20210101"
_OLSCO$(NR)="20210424":_OLSCA$(NR)="20210424"
PUTBL : IF _ERRBL<>0 THEN VIMSG 5, _ERRBL$
VIMSG 1, "Ha aggiunto una riga di PF da collegare a un OC"
IF ocnum()=290 THEN N=ocnri() '(NB: da ciclotabellare su tabella OC)
COLOCBL 25,NR,N : IF _ERRCOLOCBL<>0 THEN VIMSG 4, _ERRCOLOCBL$
VIMSG 1, "Ha collegato la riga PF all'OC | ora cancella una BL"
DELBL 22,0 : IF _ERRBL<>0 THEN VIMSG 5, _ERRBL$
```

# _PART

variabili di struttura archivi Particolarità (GET-PUT-DEL)

| | |
|---|---|
| _PARTTIP$ | (tipo particolarità P/V/S/A/I) |
| _PARTCLI$ | (codice conto) |

| | |
|---|---|
| _PARTCAC | (categoria conto) |
| _PARTART$ | (codice articolo) |
| _PARTCAA | (categoria articolo) |
| _PARTDDT$ | (data di validità da) |
| _PARTADT$ | (data di validità a) |
| _PARTIVA$ | (aliquota / esenzione) |
| _PARTVAL | (numero valuta) |
| _PARTTPF$ | (tipo fino a F/Q/P/S) |
| _PARTFIN(1-3) | (valore fino a scaglione Q,P,S - solo 3 elementi -) |
| _PARTPRZ$(1-4) | (particolarità dello scaglione _PARTFIN)<br>(indice = 4 se è Fisso, che rappresenta l'elemento 9 del video)<br>("******" se non è rappresentabile)<br>(sostituita da _PARTPRZN$ dopo l'introduzione sconti cascata) |
| _PARTTP$(1-4) | (campo "%" o "T" - solo multiagente -) |
| _PARTPROV$(1-4) | (risultato formula _PARTPRZN$ - solo multiag. e GETATTPART) |
| _PARTAGEN$(1-4) | (cod. agente - solo multiagente -) |
| _PARTCOND(1-4) | (codice condizione - solo multiagente -) |
| _PARTPRZN$(1-4) | (particolarità, a partire da implementazione sconti cascata) |
| _PARTCOD$ | (codice articolo del Cli/For se <tppart$>="A") |
| _PART_FIN(4-8) | (valore fino a scaglione Q,P,S) |
| _PART_PRZN$(4-8) | (particolarità dello scaglione) |
| _PART_TP$(4-8) | (campo "%" o "T" - solo multiagente -) |
| _PART_PROV$(4-8) | (risultato formula _PARTPRZN$ - solo multiag. e GETATTPART) |
| _PART_AGEN$(4-8) | (cod. agente - solo multiagente -) |
| _PART_COND(4-8) | (codice condizione - solo multiagente -) |

Esempio: scrive particolarità Prezzo a Quantità del cliente, le legge e necancella una e ne varia un'altra e infine, legge con GETPARTICOL.

```
AZZVARSYS 7 : _PARTTIP$="P"
_PARTCLI$="201.00028" : _PARTCAC=0 : _PARTART$="ART001": _PARTCAA=0
_PARTDDT$="20210315" : _PARTADT$="20211231" : _PARTVAL=2: _PARTTPF$="Q"
FOR I=1 TO 3 : _PARTFIN(I)=I : _PARTPRZ$(I)=STR$(I*33,0,0) : NEXT
_PARTPRZ$(4)="100": PUTPART : IF _ERRPART<>0 THEN VIMSG 4,_ERRPART$ : END
```

```
-------------------------------------------------------
Tipo particolarità.(P/S/V/A): P Prezzi
Cod.conto...: 201.00028 CNC S.r.l. CANTIERI NAVALI CLAS
Listino ven.:   ListCTO: 1 UNO
Cod.articolo: ART001 Articolo 001
Cat.form.prz:
Validità da data..: 15/03/2021
-------------------------------------------------------
Fino a data: 31/12/2021        Val.:  2 EUR EURO
Fino a: Q      Quantità        Prezzi
   1,000          33,00
   2,000          66,00
   3,000          99,00
          oltre  100,00
```

Esempio: legge Particolarità Prezzo a Quantità di un Cliente e visualizza il valore del terzo scaglione. Cancella la particolarità, la reinserisce e legge con GETPARTICOL indicando solo il conto del Cliente

```
GETPART "P","201.00028",0,"ART001",0,"20210315":IF _ERRPART<>0 THEN GOTO ERR
VIMSG 1, _PARTPRZ$(3)
DELPART "P","201.00028",0,"ART001",0,"20210315":IF _ERRPART<>0 THEN GOTO ERR

AZZVARSYS 7 : _PARTTIP$="P"
 _PARTCLI$="201.00028" : _PARTCAC=0 : _PARTART$="ART001": _PARTCAA=0
 _PARTDDT$="20210315" : _PARTADT$="20211231" : _PARTVAL=2: _PARTTPF$="Q"
   FOR I=1 TO 3 : _PARTFIN(I)=I : _PARTPRZ$(I)=STR$(I*33,0,0) : NEXT
 _PARTPRZ$(4)="100" : PUTPART : IF _ERRPART<>0 THEN VIMSG 4, _ERRPART$

AZZVARSYS 7 : IMODO=0: CONTO$="201.00028": ART$="ART001"
LOOP:
GETPARTICOL "P",COD$,CATC,ART$,CATA,"",IMODO: IF _ERRPART<>0 THEN GOTO ERR
IF _PARTCLI$="" AND _PARTART$="" AND _PARTCAC=0 AND _PARTCAA=0 THEN GOTO FIN

PRINT _PARTCLI$+" : fino a "+STR$(_PARTFIN(1),0,0)+": valore "+_PARTPRZ$(1)
PRINT _PARTCLI$+" : fino a "+STR$(_PARTFIN(2),0,0)+": valore "+_PARTPRZ$(2)
PRINT _PARTCLI$+" : fino a "+STR$(_PARTFIN(3),0,0)+": valore "+_PARTPRZ$(3)
PRINT _PARTCLI$+" :  oltre "+STR$(_PARTFIN(3),0,0)+": valore "+_PARTPRZ$(4)
VIMSG 1, "visualizzati a video|gli scaglioni della particolarità"

AZZVARSYS 7 : IMODO=1 : GOTO LOOP
VIMSG 1, "visualizzati a video|gli scaglioni della particolarità"
```

```
ERR: IF _ERRPART$<>"" THEN VIMSG 4, _ERRPART$
FIN: END
```

# _PC

variabili di struttura archivi Anagrafica Piano dei Conti (GET-PUT-DEL)

In rosso evidenziati i campi minimi obbligatori

| ANAGRAFICA | |
|---|---|
| _PCCOD$ | Cod.conto, anche "GMM.AUTO" Se parametro contabile "Rubrica unica cli/for = C" considerare il parametro contabile "All.aut.cli/for rub.unica": se = N non e' consentito l'allineamento, se = S e' consentito ma solo in forma "GMM.AUTO" non e' consentita l'assegnazione diretta del numero di conto |
| _PCIDRUBUNI | identificativo di RUBRICA UNICA di sola lettura; la PUTPC di un nuovo conto riporta l'id assegnato |
| **Annulla Riprisitna** | |
| _PCANN$ | Conto Precancellato |
| _PCCAL$ | Codice alternativo |
| **Bilancio CEE** | |
| _PCCEAV$ | Cod.CEE Avere Patr. |
| _PCCEDA$ | Cod.CEE Dare Pat/Eco |
| _PCDES$ | Descrizione conto |
| _PCDSA$ | Descr.aggiuntiva*a |
| _PCGCA$ | Giornale di cassa |
| _PCCLAS$ | Classe Conto |
| **Classificazione** | |
| _PCNBA | Num. banca su archivio BANC*a |
| _PCBASI$ | Cod. S.I.A*a |
| _PCCUC | Codice univoco CBI |
| _PCBAAZ$ | Firma Azienda*a |
| _PCBADI$ | Data iniz.rapporto*a |
| _PCBADF$ | Data fine rapporto*a |
| _PCBACC$ | n. c/corrente*a |
| _PCBATC$ | Tipo c/c*a |
| _PCBAF1 | Fido scoperto c/c |
| _PCBAF2 | Fido castelletto effetti |
| _PCBAAD$(6) | Data tasso attivo*a |
| _PCBAAL(6) | Aliquota tasso attivo*a |
| _PCBAPD$(6) | Data tasso passivo*a |
| _PCBAPL(6) | Aliquota tasso passivo*a |
| _PCBARI$ | Riferimento Signor*a |
| _PCBASE$ | Riferimento Settore*a |
| _PCBATE$ | Riferimento Telefono*a |
| _PCBANO$ | Riferimento Note*a |
| _PCASS$ | Assogettamento Iva -file rudt- |
| _PCIDT | Perc. Indetraibilita' |
| _PCALC | Aliq. di Compensazione |
| _PCRAT$ | Rateo/Risconto |
| _PCNOLOLEAS$ | Nolo leasing |
| _PCTPMANUT$ | Tipo manutenzione |
| _PCALMANUT | Percentuale manutenzione |
| _PCPAM(26) | Ammortamento standard |
| _PCMIM$ | Materiale/immateriale |
| _PCTPCESP$ | Automezzo trasportatore |
| _PCCSK$ | Conto di costo -per K- |
| _PCCSZ$ | Conto effettivo -per Z- |
| _PCIMM$ | Conto immobilizz.-per V- |
| **Conti Automatici** | |
| _PCCOR$ | C.to amm.to ordinario |
| _PCCAN$ | C.to amm.to anticipato |
| _PCCND$ | C.to amm.to non fis.ded |
| _PCAOR$ | C.to f.do ordinario |
| _PCAAN$ | C.to f.do anticipato |
| _PCAND$ | C.to f.do non fisc.ded |
| _PCPLU$ | C.to plusvalenze |
| _PCPND$ | C.to plusv.non fisc.tas |
| _PCMIN$ | C.to minusvalenze |
| _PCMND$ | C.to minus.non fis.ded |
| _PCIRPE | obsoleta, non utilizzabile |
| _PCIRAP | obsoleta, non utilizzabile |
| _PCNAZ$ | Nazionalita' |
| _PCPAE$ | Paese |
| _PCPRI$ | Privato |
| _PCPFS$ | Pers.fisica |
| _PCPAENASC$ | Paese di nascita - black list - [STATN] |
| _PCDTNASC$ | Data di nascita - black list - [STATN] |
| _PCCOG$ | Cognome |
| _PCNOM$ | Nome |
| **Ragione sociale alla data** | |
| _PCDAD$ | Fino alla data |
| _PCNAD$ | Ragione Sociale |
| _PCRIC$ | Nome di ricerca |
| **Generale** | |
| _PCDOCRIEP$ | documento riepilogativo per spesometro 'S'/'N' |
| _PCSRA | Soggetto ritenuta acconto |
| _PCDOMU$ | Condominio |
| _PCNOSPESOM$ | Escludi spesometro N/S |
| _PCAGRI34C6$ | Agricolo A34 c6 N/S |
| _PCENTEP$ | Ente Pubblico |
| _PCESPLITPG$ | Escludi da gestione Split Payment |
| _PCCTR$ | Cod.tributo |
| _PCCATASTD$ | cod. catastale domicilio percipiente,'F'cod.trib./n.iniziat. |
| _PCMANI$ | Att.Marg./Nuove Iniz. |
| _PCDMANI$ | Data validita' _PCMANI$ |
| _PCFDP | % f.do professionista |
| _PCTPRE | Tipo prestazione |
| _PCGLAD$ | GLA/D |
| _PCALCOL | % coll.coo.-vend.porta/porta |
| _PCALDFT | % Deduzione forfetaria |
| _PCATCOL | Codice attivita' |

| _PCDTN$ | Data nascita/costituz. |
|---|---|
| _PCLON$ | Localita' nascita/costituz. |
| _PCPRN$ | Provincia nascita/costituz. |
| _PCIND$ | Indirizzo |
| _PCCAP$ | CAP |
| _PCLOC$ | Localita' |
| _PCPRO$ | Provincia |
| _PCTEL$ | Telefono |
| _PCFAX$ | Numero fax |
| _PCINT$ | Indirizzo E-Mail |
| _PCINT1$ | Indirizzo Internet |
| _PCCFI$ | Codice Fiscale |
| _PCNPI$ | Partita Iva |
| _PCPIS$ | P.Iva II^ |
| _PCNOPIVA$ | Senza P.Iva S/N; Italiani, noprivati e cf=solonumerico |
| _PCCODUPA$ | Codice Univoco Ufficio Pubblica Amministrazione |
| _PCFATELEPA$ | Fattura elettronica P.A. S/N |
| _PCRIFAMMPA$ | Riferimento amministrazione PA |
| _PCOBRFAMPA$ | Obbligo Riferimento documenti |
| _PCSPORTIVO$ | Collaboratore sportivo S/N |
| _PCSPORTAZI$ | Collaboratore esclusivo S/N |
| _PCSPORTALR | Aliquota addizionale regionale collaboratore sportivo |
| _PCSPORTALC | Aliquota addizionale comunale collaboratore sportivo |
| **Condizioni commerciali** | |
| _PCPAG | Pagamento abituale |
| _PCSCF | Scadenza fissa |
| _PCSPC | Giorni di proroga |
| _PCMES(2) | Mese escluso |
| _PCMAP(2) | Giorno antic/post. |
| _PCBAP$ | Banca d'appoggio |
| _PCCCB$ | Cod. conto corrente |
| _PCBBCIN$ | Cod. Cin su BBAN*a |
| _PCABI | Codice ABI |
| _PCCAB | Codice CAB |
| _PCCOB$ | Cod banca present |
| _PCCST | Categoria statistica |
| _PCONA | Zona |
| _PCREF$ | Raggruppam. effetti S/N/P/U/Y U=S+spese 1^ fattura; Y=P+spese banca su 1^ fattura |
| _PCAGE$ | Cod.conto agente |
| _PCAGECOND | Condizioni agente |
| _PCLIS | Listino |
| _PCLIE$ | Extra listino |
| _PCVAL | Valuta estera |
| _PCLIN | Lingua estera |
| _PCCSC | Categoria sconti |
| _PCCPR | Categoria provvigioni |
| _PCASI$ | Assogettamento Iva -file pico- |
| _PCRBL$ | Raggruppam. documenti |
| _PCIEV$ | Docum. interamente evadibile |
| _PCBES$ | Bollo esportatore abituale |
| _PCFTA$ | Codice conto fatturare a |
| _PCPARDEST$(10) | Condizioni destinatario |
| _PCSCO$ | Sconto incondizionato |
| _PCSPEDIZ$ | Cod.anagraf.indiriz. spediz. |
| _PCCIGCUP$ | Gestione CIG-CUP abilitato S/N |
| _PCCONTRSPS$ | Tipo contratto spesometro: N/C/P |
| _PCPAGA | Pag.art. alimentari non deperib. |
| _PCPAGD | Pag.art. alimentari deperibili |
| _PCOPTURIS$ | Spesometro OP.legate al Turismo |
| **Enasarco** | |
| _PCENAINIRA$ | Data inizio rapporto, se e' presente significa che il Soggetto Enasarco e' = S |
| _PCENAFINRA$ | Data fine rapporto |
| _PCENAVARRA$ | Data variazione rapporto |
| _PCENATPATT$ | Tipo mandato attuale |
| _PCENAIMPTR(3,4) | Progressivo imponibile, sola lettura |
| _PCENAIMPRT(3,4) | Rettifica imponibile |
| _PCENATPMAN$(4) | Tipo mandato corrispondente, sola lettura |
| _PCENACONTR(4) | Contrib. calc. da documenti, sola lettura |
| _PCENADOVUT(4) | Contrib. dovuto con compens.,sola lettura |
| _PCENAINTEG(4) ( | Integrazione dovuta, solo lettura Nota per la dimensione a 3 elementi:3=ultimo anno; 2=penultimo anno; 1=non attivo |
| **Contropartite** | |
| _PCCTRPRC$ | Cod.Ricav.stand.raggruppam. |
| _PCCTRPPC$ | Cod.Controp Ric.personale |
| _PCCTRPMA$ | Cod.Controp C/F artic'A' |
| _PCCTRPML$ | Cod.Controp.C/F artic.'L' |
| _PCCTRPMP$ | Cod.Controp C/F artic'P' |
| _PCCTRPMS$ | Cod.Controp.C/F artic.'S' |
| _PCCTRPMZ$ | Cod.Controp C/F artic'Z' |
| **Coordiante bancarie Estese** | |
| _PCCDRID$ | Codice cliente su RID |
| _PCBIC$ | Codice di BIC, Condizioni*a |
| _PCIBPAE$ | Cod.paese per codif IBAN*a |
| _PCIBCIN | Cod. cin per codif IBAN*a |
| _PCIBCOD$ | Cod libero su codif.IBAN*a |
| _PCIBAN$ | C.IBAN -IBPAE+IBCIN+IBCOD- |
| _PCBBAN$ | Codice BBAN -cin+ABI+CAB+c/c- |
| _PCRIDTPI$ | tipo incasso RId "" o "V" sdd B2B |
| _PCRIDIDE$ | codice identificativo impresa RID arricchito conti classificati banca |
| _PCRIDDTM$ | data mandato rid clienti banca standard |
| _PCRIDSTS$ | stato sequenza rid clienti banca stand. |
| **Ulteriori banche d'appoggio** | |
| _PCUBBAP$(3) | Descrizione |
| _PCUBRIDDTM$(3) | Data mandato rid clienti |
| _PCUBRIDSTS$(3) | Stato sequen.rid clienti |
| _PCUBRIDTPI$(3) | Tipo di incasso "" o V |
| _PCUBBIC$(3) | Codice BIC della banca |
| _PCUBPAE$(3) | Sigla paese IBAN a |
| _PCUBIBCIN(3) | Cin IBAN IBAN b |
| _PCUBIBCOD$(3) | Codice IBAN IBAN c = BBAN |
| _PCUBBACIN$(3) | Cin BBAN BBAN 1 |
| _PCUBABI(3) | Codice ABI BBAN 2 |

| | |
|---|---|
| _PCUBCAB(3) | Codice CAB BBAN 3 |
| _PCUBCCB$(3) | C. corrente BBAN 4 |
| _PCUBIBAN$(3) | Codice IBAN solo GET a+b+c |
| _PCUBBBAN$(3) | Codice BBAN solo GET 1+2+3+4 |
| _PCUBCODRID$(3) | Codice Cliente x RID |
| **RID SEPA** | |
| _PCSDDMND | Elemento mandato su banca appoggio base |
| _PCUBSDDMND(3) | Elemento mandato ulteriori banca appoggio |
| _PCSDDDATMA$(5) | Data mandato |
| _PCSDDCODI$(5) | Numero/codice mandato |
| _PCSDDSTAT$(5) | Stato mandato |
| _PCSDDTIPO$(5) | Tipo mandato |
| **Fattura Elettronica** | |
| _PCFATTELE$ | Fattura elettronica |
| _PCSERIELE | Serie Fatt. elettronica |
| **Recapito** | |
| _PCTPRECAP2$ | Tipologia recapito P,A,L |
| _PCNOME2$ | Nominativo |
| _PCIND2$ | Indirizzo |
| _PCCAP2$ | CAP |
| _PCLOC2$ | Localita' |
| _PCPROV2$ | Provincia |
| _PCINDEFF$ | Riporta su effetti N,S,R,E |
| _PCTPSOC$ | Tipo societa' N,I,C,P |
| **Vettore** | |
| _PCVET$ | Cod.conto vettore |
| _PCVPO$ | Porto |
| _PCVDE$ | Descrizione porto |
| _PCINCOT$ | Codice Incoterm |
| _PCVSP$ | Tip.spese di spedizione |
| _PCVVA | Valore spese di spediz. |
| _PCNISCALBO$ | Iscrizione all'albo |
| _PCDISCALBO$ | Data di iscrizione all'albo |
| **Imballi** | |
| _PCIMB(16) | Quantita' cauzioni |
| _PCVAC$ | Valorizza cauzioni |
| **Progressivi / Saldi** | |
| _PCINS | Numero insoluti |
| _PCVIN | Valore insoluti |
| _PCFID | Fido migl.Euro, mil.Lire |
| _PCFIDO | Fido valuta di gestione se fido e' < 0 significa % su fatturato anno precedente |
| _PCFID | prevale su _PCFIDO -azzerarla prima di usare _PCFIDO- |
| _PCFIDTPMSG$ | Msg. fuori fido emissione doc. N/A/B |
| **Dati Parcella** | |
| _PCRSP$ | Responsabile |
| _PCSPS$ | Spese anticipate studio |
| _PCPRE$(12) | Cod.-Descriz. prestaz.'R' |
| _PCPQT(12) | Quantita' |
| _PCPMS(12) | Mese |
| _PCPPR(12) | Prezzo |
| _PCAZI$ | Sigla azienda del cliente |
| _PCCPF$ | Contab. parcelle -def.N-*b |
| _PCSTU$ | Cod. fornit. studio |
| _PCOST$ | Cod. costo prestazione cli. |
| _PCCPG$ | Registra PG di FR -def.N-*b |
| _PCPPG$ | Conto da usare per il PG*b |
| **Note** | |
| _PCNDS$(2) | Descrizione nota |
| _PCNDE$(2) | Destinazione nota |
| _PCNDT$(2) | Data di scadenza nota |
| **Videate** | |
| _PCVDR$(26) | 26 campi videata anagraf. riportabile*a |
| _PCVDF$(26) | 26 campi videata anagraf. non riportabile*a |
| _PCWWW$(26,24) | 26 videate aggiuntive con 24 campi la GET converte da num->str$, la PUT viceversa |
| **Stato anagrafica** | |
| _PCDCR$ | Data creazione conto |
| _PCDUM$ | Data modifica conto |
| _PCFMO$ | Conto modificato S/N |
| _PCMRA$(12) | Stato anagr.attiva; v.mirroring*a |
| _PCMRV$(12) | Stato anagr.variata; v.mirroring*a |
| **Spesometro** | |
| _PCRLCOGNO$ | Cognome del Rappresentante legale |
| _PCRLNOME$ | Nome del Rappresentante legale |
| _PCRLDTN$ | Data nascita del Rappr. legale |
| _PCRLLON$ | Loc. nascita del Rappr. legale |
| _PCRLPRN$ | Prov. nascita del Rappr. legale |
| _PCRLPAE$ | Codice Domicilio fiscale Rapp. leg. |
| **Dati sistema** | |
| _PCDTSIST$ | Data sistema ultima scrittura fisica |
| _PCORSIST$ | Ora sistema ultima scrittura fisica |
| **Controllo di Gestione** | |
| _PCCRIRIPCF | Numero del criterio di ripartizione per Clienti/Fornitori |
| _PCANLCF$ | Codice analitico per Clienti/Fornitori |
| _PCCRIRIP | Numero del criterio di ripartizione per conti diversi da C/F |
| _PCANL$ | Codice analitico per per conti diversi da C/F |

*Note:*

*a - variabili condizionate dalla inizializzazione di _PCSOLOPICO$="S/N",
(vedere le indicazioni riportate nella sezione della PUTPC).
Notare: in presenza di raggruppamento, la possibilita' di identificare conti come di tipo banca, non passa piu' solo dall'attribuzione di _PCCLAS$="B" (che distingueva rispetto a conti di tipo Cassa, per esigenze di flussi finanziari), ma passa anche dal tipo di Mastro, per il quale e' attiva l'attribuzione come "Depositi Bancari". Se infatti nel raggruppamento, un Mastro patrimoniale appartenente a un gruppo visibile in Anagrafica Piano dei Conti viene marcato come "Ban" (cioe'come Deposito Bancario), questo Mastro (e relativi conti) sara' visualizzato in Rubrica clienti/fornitori/banche, perche' appartenenti alla specifica azienda in questi one. Se poi il raggruppamento e su piano dei conti riservato (80/81), i conti non riservati (quindi personalizzabili) sono quelli con mastro superiore al 69.
*b - contabilizzazione parcelle per aziende studio con Collage;Esempio: copia di conto, attribuzione di P.Iva/C.F., controllo+cancellazione

```
AZZVARSYS 1: GETPC "201.00007": IF _ERRPC<>0 THEN VIMSG 5, _ERRPC$: END
_PCCOD$="201.AUTO": _PCDES$="Copia Conto": _PCNAZ$="I": _PCLIS=1: _PCVAL=2
_PCNPI$ = "12345678900" : _PCCFI$ = "CPACNT82A03H294J"
PUTPC: IF _ERRPC$<>"" THEN VIMSG 4, _ERRPC$
```

```
PI = CTRPIVA(panpi("201.00177"))  'oppure PI = CTRPIVA("12345678900")
IF PI = 0 THEN VIMSG 1, "P.Iva errata" ELSE VIMSG 1, "P.Iva corretta"
CF = CTRCODFIS(pacfi(_PCCOD$))
IF CF = 0 THEN VIMSG 1, "C.F. errato" ELSE VIMSG 1, "C.F. corretto"
DELPC _PCCOD$ : IF _ERRPC<>0 THEN VIMSG 5, _ERRPC$ : END
```

# _PN

variabili di struttura archivi Primanota (GET-PUT-DEL)

| _PNOPE | Numero operatore, 0=AUTO |
|---|---|
| _PNSAZ | Codice sottoazienda -solo lettura-, v. APRIAZIE |
| _PNPRN | Numero progressivo -solo lettura- v. <prinr> |
| _PNDRE$ | Data registrazione |
| _PNCAU$ | Causale contabile |
| _PNTDE$ | Descrizione di testata |
| _PNCCR(1) | Centro costo/ricavo - v. anche _PNCCR di riga |
| _PNRPR$ | Registro protocollo Iva |
| _PNSPR | Numero serie protocollo |
| _PNNPR | Numero protocollo, 0=AUTO |
| _PNNDO | Numero documento, 0=AUTO |
| _PNNUMRE$ | Alfanumerico documento - NB: vedere note variabile _MMNUMRE$ |
| _PNDDO$ | Data documento |
| _PNVRP$ | Versione Documento se prima della 2010G torna "S" |
| _PNPVEN | Punto Vendita |
| _PNDST$ | Data statistica |
| _PNDTD$ | Data decorrenza pagamento |
| _PNDFP$ | Data pag.Fatt.esig.diff. |
| _PNRIT$ | Flag registraz. Iva Art.27 |
| _PNDRV$ | Data rif. vidimazione uff.tribut. RSM |
| _PNPAG | Cod. pagamento; solo PUT per generare rate scadenzario |
| _PNPCA$ | Causale doc. in pagamento |
| _PNPSD | Serie docum. in pagamento |
| _PNPND | Numero docum. in pagamento |
| _PNPDD$ | Data docum. in pagamento |
| _PNTDER$ | Nota registro [F4] Testata |
| **PROTOCOLLO VENDITE** | |
| _PNRPS$ | Registro |
| _PNSPS | Numero serie |
| _PNNPS | Numero documento |
| _PNINT$ | Intra CEE(S/N |
| _PNVAL | Codice valuta estera |
| _PNDSV$ | Descrizione valuta |
| _PNEST | Importo valuta |
| _PNCMR | Codice mandato RID SEPA |
| **Variazione Nota** | |
| _PNNVA$ | Nota variazione anni precedenti [F6] |
| **Categoria Bilancio** | |
| _PNBIL$ | Categ.bilancio specializzato [SHF6] |
| **VALUTA DOCUMENTO** | |
| _PNVDO | Valuta documento*a |
| _PNEVE + _PNDIE | Cambio valuta Doc.->Euro* + (Flag cambio 0=No/1=Si)*a |
| _PNEVC + _PNDIC | Cambio Euro->valuta Gest* + (Flag cambio 0=No/1=Si)*a |
| **BLACk LIST** | |
| _PNDCOBL$ | Periodo di competenza |
| _PNIOPBL$ | Includi operazioni nel elenco |
| _PNNVABL$ | Nota di variaz. operazioni del anno |
| _PNNVPBL$ | Nota di variaz. operazioni anni preced. |
| _PNCTRBL$ | Codice cliente blacklist |
| _PNIEPBL$ | Escludi operazioni blacklist |
| **SPLIT PAYMENT** | |
| _PNILSP$ | Includi in liquidazione operazione Split Payment |
| **IVA MOSS** | |
| _PNNPPM$ | (S/N) Nota credito riferita a trimestri precedenti |
| **RIGHE DOCUMENTO** | |
| _PNCTO$(200) | Codice conto |
| _PNCCR(200) | Centro costo/ricavo |
| _PNDES$(200) | Descrizione |
| _PNIMD(200) | Importo contabile*a |
| _PNMDO(200,4) | Imponibile iva*a |
| _PNIVD(200,4) | Imposta*a |
| _PNIMP(200) | Importo contabile |
| **CASTELLETTO IVA** | |
| _PNIMB(200,4) | Imponibile iva |
| _PNIVA(200,4) | Imposta iva |
| _PNALI$(200,4) | Aliquota iva |
| _PNALD$(200,4) | Aliquota iva ventil. diff. |
| _PNCND$(200,4) | Castelletto no doppio protocollo 'S'/'N' |
| **SPESOMETRO** | |
| _PNEDPE$ | Escludi docum. per pagam. elettr.'S'/'N' |
| _PNFCS$ | Forza comunicaz. se doc. inf. limite F/P/N |
| _PNDIMU$ | Documento importo unitario 'A'/'S'/'N'*d |
| _PNIMTOSP | Importo totale dell'operazione*d |
| _PNIITOSP | Imposta su totale operazione*d |
| _PNCIC$() | Cointestatari spesometro - conto |
| _PNCIP() | Cointestatari spesometro - percentuale |
| _PNNVR$(200,4) | Nota di variazione spesometro 'S'/'N' |
| _PNRSVA(200) | Rif. docum. serie per nota di variazione |
| _PNRNVA(200) | Rif. docum. numero per nota di variazione*d |
| _PNRDVA$(200) | Rif. docum. data per nota di variazione |
| _PNPSER$(200) | Prestazione di servizio*d |
| _PNORCO$(200) | Operazione riferita a contratto 'S'/'N' |
| _PNECAC$(200) | Escludi castelletto da comunicazione 'S'/'N' |
| _PNCFPI$(200) | Codice fiscale / Partita Iva |
| _PNCCS$(200) | Codice cliente |
| _PNCCTR$(200) | Codice contratto; se non sono previste istruzioni sull'archivio contratti, non e'controllata l'esistenza del contratto |

| _PNROPE$(200) | Riferimento operazione |
|---|---|
| _PNFRIMP(200) | Importo fattura da rettificare imponibile*d |
| _PNFRIMPO(200) | Importo fattura da rettificare imposta*d |
| _PNFCS$(200) | Forza comnunicazione se doc. infer.al limite |
| _PNRDN$(200) | Rifer.doc.numero esteso nota di variazione |
| _PNTPC$(200) | Tipo contratto |
| _PNTNL$(200) | Tipo noleggio leasing |
| _PNOTU$(200) | Op.legata al turismo |
| **RIFERIMENTI PAGAMENTI PS** | |
| _PNEDC$(200) | Codice conto |
| _PNEDS(200) | Ser. doc.in pagamento |
| _PNEDN(200) | Num. doc.in pagamento |
| _PNEDD$(200) | Data doc.in pagamento |
| **CONTABILITÀ ANALITICA** | |
| _PNSCO$(200,50) | Codice analitico |
| _PNSCM$(200,50) | Commessa |
| _PNSAN(200,50) | Anno |
| _PNSIM(200,50) | Importo |
| **CONTABILITÀ ANALITICA (CDG)** | |
| _PNGCO | Coll.An. 0=no 1=si*b |
| _PNSCOz(200,15) | Codice analitico*b |
| _PNSCM$(200,15) | Commessa*b |
| _PNARE$(200,15) | Codice area*b |
| _PNDRI$(200,15) | Data riferimento*b |
| _PNQTA(200,15) | Quantita'*b |
| _PNSIM(200,15) | Importo*b |
| **DATI INTRACEE** | |
| _PNAOP(50) | Ammontare |
| _PNNTR$(50) | Natura transazione |
| _PNNCM(50) | Nomenclatura combinata |
| _PNAOV(50) | Ammontare operazioni in valuta |
| _PNMNE(50) | Massa netta |
| _PNUSP(50) | Unita' suppletiva |
| _PNVST(50) | Valore statistico |
| _PNRGM$(50) | Condiz.consegna -ex regime statis- |
| _PNMTR$(50) | Modalita' trasporto |
| _PNPDE$(50) | Paese destinazione/provenienza |
| _PNPPO$(50) | Provincia/paese origine |
| _PNPDS$(50) | Provincia destinazione |
| _PNIEU$ | Importi Intra in Lire("N")Euro("S") sola lettura; automatica da PUT, se anno < a 2001 o valuta di gestione in LIRE e' impostata a "N" altrimenti "S" |
| **DATI INTRA Sez.2** | |
| _PNMES(50) | Mese di riferimento 1-12 |
| _PNTRI(50) | Trimestre di riferimento |
| _PNANN(50) | Anno di riferimento |
| _PNSTA$(50) | Codice iso paese |
| _PNCIV$(50) | Codice iva del conto |
| _PNSGN$(50) | Segno degli importi |
| _PNNAT$(50) | Natura transitoria |
| _PNAOE(50) | Ammontare op. in Euro |
| _PNAVA(50) | Ammontare op. in valuta |
| _PNVAS(50) | Valore statistico |
| _PNNOC(50) | Nomenclatura combinata |
| **PROVVIGIONE** | |
| _PNAGE$ | Codice Agente |
| _PNPRA$(15) | Codice Agente |
| _PNPRO | Importo provv. complessivo |
| _PNPRI(15) | Importo provv. complessivo |
| _PNPRP | Importo provv. pagato |
| _PNPRG(15) | Importo provv. pagato |
| **RITENUTA D'ACCONTO** | |
| _PNCTR | Cod. tributo Rda [n]NNNN, se di 5 cifre Ritenuta d'Acconto [n] e'indice tab.al.Iva associabile a trib |
| _PNDPT$ | Documento totalmente pagato; valori S/N; Ritenuta d'Acconto se = N l'assenza dello specifico conto automatico 'Erario c/rit. provvisorio' blocca la registrazione PUTPN |
| _PNIRA | Importo Iva Rda |
| _PNNS1 | Imponib. non sog. Rda |
| _PNNS2 | Spese non sog. Rda |
| _PNFNS | Fondo prof. non sogg.a Rda |
| _PNSOG | Imponibile Rda |
| _PNARA | Aliquota Rda |
| _PNRIT | Importo Rda |
| _PNDSC$ | Descrizione Rda |
| **ENASARCO.** | |
| _PNENA | Importo ENASARCO |
| _PNENZ | Importo enasarco a carico dell'azienda |
| _PNDCE$ | Data competenza Enasarco |
| **INPS CO.CO.CO.** | |
| _PNRPR | Rit. previdenz.1/3 |
| _PNIMA | Imponibile sogg.contr al netto deduzione |
| _PNPED | Percentuale aliquota contributo |
| _PNPDF | Percentuale deduzione forfetaria |
| _PNIMC | Importo contributo |
| _PNIAC | Importo contr. a carico committente 2/3 |
| _PNPAD$ | Periodo attivita' dal |
| _PNPAA$ | Periodo attivita' al |
| _PNCOA | Codice attivita' collaboratore |
| _PNDPC$ | Data pagamento compenso |
| _PNIDV | Importo contributo da versare |
| _PNDVE$ | Data versamento |
| _PNIMS | Imponibile soggetto contributo |
| _PNPRS | Tipo prestazione -tabella TPRE- |
| **SCADENZARIO A PARTITE** | |
| _PNPAR$ | Partita Generale*c |
| _PNSCT$(249) | Cli/For*c |
| _PNSPA$(249) | Partita*c |
| _PNSDS$(249) | Data Scadenza*c |
| _PNSID(249) | ID Scadenza*c |
| _PNSTP$(249) | Tipo pagamento rata*c |
| _PNSCA$(249) | Causale documento*c |
| _PNSSE(249) | Serie docum. Iva*c |
| _PNSNU(249) | Numero documento*c |
| _PNSDD$(249) | Data documento*c |

| _PNSIP(249) | Importo pagamento*c |
|---|---|

*Note:*

*a - solo GETATTPN con valuta documento <> da quella di gestione contabile: se il documento è emesso nella stessa valuta di gestione dell'azienda il Cambio = 1 e Flag = 0. Dato un importo in valuta di partenza il Flag stabilisce come operare per calcolare l'importo nella valuta di arrivo: se Flag= 0 moltiplicare oppure se = 1 dividere. Es. triangolazione con area Euro (100$->Eu-Lir) dati i cambi $ : Eu = 1,1565:1 e Eu : Lir = 1:1936,27 : (se _PNDIE=1 e _PNEVE=1,1565 -> 100:1,1565 = 86,47 Eu; (se _PNDIE=0 e _PNEVE=0,86467 -> 100*0,86467 = 86,47 Eu -cambio inverso-; Eu -> Lir (_PNDIC=0 e _PNEVC=1936,27) 87,47 Eu*1936,27 = 167,429 Lir

*b - obsolete e sostituite da specifiche _CDG? del Controllo di Gestione, caricabili comunque da GETATTPN. Gli archivi di _CDG/_PN/_MM sono separati ma collegati (es. conto di riga, progressivi di documento/riga). GETATTPN carica tutte le possibili _CDG ma questo array monodimensionale è distribuibile in _PN (array bidimensionale) con ciclo sul progressivo di riga: ON_612_ALL: GETATTPN : FOR X=1 TO 4 PRINT _CDGSCODANL$(X)+"|"+_CDGCTOCG$(X)+"|"+STR$(_CDGPROGRIG(X),0,0)+"|"+_PNCTO$(_CDGPROGRIG(X: NEXT

*c - solo scadenzario a partite: _PNPAR$ èla PARTITA (variabile generale di 13 caratteri alfanumerici non leggibili con GET, esempio "04FE01/000077")da assegnare a tutte le rate all'inserimento in scadenzario di nuovi documenti; se impostata, sostituisce quella automatica generata dal programma secondo i criteri di primanota (sia per importi positivi, che negativi); la selezione delle 1 - 249 rate obbliga l'indicazione del conto _PNSCT$ e può effettuarsi in base ai seguenti schemi di pagamento delle rate per: PARTITA, RIFERIMENTI DOCUMENTO, DATA SCADENZA e/o TIPO DI PAGAMENTO.

-> _PNSCT$+_PNSPA$+_PNSDS$+_PNSID+_PNSTP$+_PNSIP

-> _PNSCT$+_PNSCA$+_PNSSE+_PNSNU+_PNSDD$+_PNSDS$+_PNSID+_PNSTP$+_PNSIP

-> _PNSCT$+_PNSDS$+_PNSID+_PNSTP$

Il _PNSTP$="M" è Rimessa, ="" è tutti; _PNSID è un progressivo da 0 a 255(se > 0 obbliga l'indicazione di _PNSDS$ altrimenti viene ignorato) e va impostato solo se nella stessa partita di un cli/for ci sono più scadenze nello stesso giorno. L'array precedentemente indicato (1-249) infatti non significica infatti che l'indice rappresenta la singola rata, ma serve a identificare ciò che si vuole pagare (più conti di uno stesso documento, più documenti ecc.fino a un massimo di 249 operazioni di pagamento, ovvero il limite fisicamente raggiungibile). L' uso dei RIFERIMENTI esclude la ricerca per partita (se impostata vengono ignorati). Sull'operazione dei pagamenti vengono riportati in automatico i vari dati (Partita, Scadenza+ID.,...) delle rate pagate (ereditati). I pagamenti si possono eseguire impostando i campi di selezione: se questi non vengono valorizzati viene generata una operazione di scadenzario per ogni cli/for presente nell'operazione di primanota, pari al relativo saldo contabile (come partita è assunta quella automatica generata da programma). ATTENZIONE: questo importo è considerato solo se nel criterio di pagamento si specifica la Partita oppure i Riferimenti Documento, altrimenti viene ignorato. L'importo che guida l'operazione è sempre quello di primanota:un pagamento non può essere > al valore della rata pagata, e la somma degli importi immessi per ogni cli/for deve sempre quadrare col relativo saldo di primanota (somma degli importi _PNIMP di _PNCTO$); in ogni caso, se la somma degli importi immessi è > al saldo di primanota viene eseguito il pagamento solo per l'importo corrispondente al saldo di primanota del cli/for (la parte eccedente viene ignorata); se la somma è < viene aggiunta nello scadenzario del cli/for una rata = alla differenza (la partita assunta è la variabile generale e se assente, quella automatica generata dal programma). Se l'importo pagamento non è impostato sono pagate tutte le rate del cli/for immesso in base ai campi di selezione Partita, Scadenza, Tipo Pagamento, anche in combinazione fra loro (AND). Se non è indicata la data scadenza i pagamenti vengono effettuati ad iniziare dalla rata più vecchia fino a raggiungere il saldo contabile di primanota del cli/for (fino a 249 rate per ogni cli/for).

Le rate dei documenti con Ritenuta D'Acconto (max 24) vanno interamente pagate (se pagata parzialmente la rata viene spezzata e per il residuo è creata automaticamente una specifica rata). Lo Scadenzario a Partite consente 250 rate per ogni cliente/fornitore esistente nelle singole operazioni di primanota, 24 rate per documenti con RDA, 255 rate nella stessa data scadenza per ogni partita dei singoli clienti/fornitori. L'array è limitata a 249 elementi poichèl'ultimo elemento e riservato al programma per gestire la quadratura del saldo contabile del cli/for presente nell'operazione di primanota con il saldo degli importi pagati dallo stesso cliente/fornitore.

*d - Variabile obsoleta e non più attiva dalla versione v550 (2011F)

Esempio:

revisione fattura emessa (non pagata) con Scadenzario a Documento; la revisione prevede contestuale pagamento del documento. Successivamente, una registrazione simile alla revisione viene eseguita su uno Scadenzario a Partite, del quale viene pagata una singola rata.

```
-----
Data: 03/01/21 Valuta: EUR |CNC S.r.l. CANTIERI NAVALI
Caus: FE   FATT. EMESSE    |Saldo precedente..:   23.700,00D
Desc: Descrizione FE C.N.C.    Co/ric:   1|
Prot: V01/000141 Ndoc:   141 Del: 03/01/21|
001 Codice Descrizione conto Importo (EUR)|Doc.Num. Scadenza Tipo   Importo
201.00028 CNC S.r.l. CANTIERI       23700D|FE   14103/02/21M      3.700,00E
803.00001 MERCI C/VENDITE       11.700,00A|FE   14103/03/21M      5.000,00E
805.00001 RICAVI DA PRESTAZ.     9.090,91A|FE   14103/04/21M      5.000,00E
341.00052 IVA VENDITE            2.909,09A|FE   14103/05/21M     10.000,00E
-----
      castelletto Iva 803.00001     |      castelletto Iva 805.00001
-----
    Aliq. Imponibile  Imposta       |    Aliq. Imponibile  Imposta
```

```
    20     10.000,00  2.000,00      |      10       9.090,91  909,09
    N74,3   1.700,00  N.I.art.74/c  |
-----
```

```
AZZVARSYS 2:
GETPN"20210103",0,"FE","V",1,141,141,"20210103" : IF _ERRPN<>0 THEN GOTO ERR
DELPN"20210103",4236,"","",0,0,0,"" : IF _ERRPN<>0 THEN GOTO ERR

 _PNDRE$="20210406": _PNRPR$="V": _PNSPR =1        : _PNNPR=0
 _PNCAU$="FE01"    : _PNNDO =0  : _PNDDO$="20210406" : _PNPAG=0 '(rimessa)

 _PNCTO$(1)  = "201.00029" : _PNIMP(1)  = 1200
 _PNCTO$(2)  = "803.00001" : _PNIMP(2)  = -440
 _PNALI$(2,1)= "20"       : _PNIMB(2,1)=  300   : _PNIVA(2,1) = 60
 _PNALI$(2,2)= "N74.3"     : _PNIMB(2,2)=  140   : _PNIVA(2,2) = 0
 _PNCTO$(3)  = "805.00002" : _PNIMP(3)  = -636.36
 _PNALI$(3,1)= "10"       : _PNIMB(3,1)=  636.36 : _PNIVA(3,1) = 63.64
 _PNCTO$(4)  = "341.00052" : _PNIMP(4)  = -123.64

PUTPN : IF _ERRPN<>0 THEN GOTO ERR
AZZVARSYS 2
GETPN "20210406",_PNPRN,"","",0,0,0,"" : IF _ERRPN<>0 THEN GOTO ERR
 NUM = _PNNPR : DAT$ = _PNDRE$
AZZVARSYS 2 '(altrimenti nel pagamento sarebbero presenti anche conti Iva)

 _PNCAU$   ="PG"      : _PNDRE$  =DAT$
 _PNPCA$   ="FE"      : _PNPSD   =1     :_PNPND=NUM :_PNPDD$="20210406"
 _PNCTO$(1)="201.00029" : _PNIMP(1)=-1200
 _PNCTO$(2)="324.00001" : _PNIMP(2)= 1200    '(324.00001 è conto cassa)

PUTPN : IF _ERRPN<>0 THEN GOTO ERR
ERR: VIMSG 4, _ERRPN$
```

Nella primanota con scadenzario a partite viene registrata la seguente FR e si paga la seconda rata di una partita con 3 rate di rimessa diretta dalla data della fattura a fine mese.

```
Data: 03/01/07 Valuta: EUR |ACCIAI srl T
Caus: FR   FATT. RICEV.    |Saldo precedente..:    4.630,00A
Desc:      Co/ric:    |Imp. da scadenzare:          0,00
Prot: A01/000007 Ndoc:     3 Del: 03/01/07|
001 Codice Descrizione conto Importo (EUR)|P a r t i t a           Saldo
601.00008 ACCIAI srl             1.200,00A|07FR01/000001            430,00E
702.00051 MERCI C/ACQUISTI         440,00D|07FR01/000002          4.000,00E
702.00024 IMBALLAGGI C/ACQUI       636,36D|07FR01/000003          1.200,00E
212.00001 IVA SU ACQUISTI          123,64D|
         Partita: 07FR01/000003
       ----------------------------
        Scadenza           Saldo
        28/02/07           400,00E
        31/03/07           400,00E
        30/04/07           400,00E
-----
      castelletto Iva 702.00051     |      castelletto Iva 702.00024
-----
    Aliq. Imponibile  Imposta       |    Aliq. Imponibile  Imposta
    20        300,00     60,00      |     10         636,36   63,64
    N38,4     140,00  N.I.art38qut  |
-----
  PARTITA: 07FR01/000003        Saldo:     1.200,00E
Documento      del    Scadenza Tipo      Importo  Dt.Pag/Rif SaldoScadenza
FR 1/     3 03/01/07 28/02/07 M          400,00E   400,00E
FR 1/     3 03/01/07 31/03/07 M          400,00E   400,00E
FR 1/     3 03/01/07 30/04/07 M          400,00E   400,00E
-----
```

```
AZZVARSYS 2:
 _PNDRE$="20070103":_PNRPR$="A":_PNSPR =1        :_PNNPR=0
 _PNCAU$="FR"    :_PNNDO = 3  :_PNDDO$="20070103"
_PNPAG=9  '(rimessa 30/60/90 data fattura fine mese)
 _PNCTO$(1)  = "601.00008" : _PNIMP(1)  = -1200
 _PNCTO$(2)  = "702.00051" : _PNIMP(2)  =   440
 _PNALI$(2,1)= "20"       : _PNIMB(2,1)=   300   : _PNIVA(2,1) = 60
 _PNALI$(2,2)= "N38.4"     : _PNIMB(2,2)=   140   : _PNIVA(2,2) = 0
 _PNCTO$(3)  = "702.00024" : _PNIMP(3)  =   636.36
 _PNALI$(3,1)= "10"       : _PNIMB(3,1)=   636.36 : _PNIVA(3,1) = 63.64
 _PNCTO$(4)  = "212.00001" : _PNIMP(4)  =   123.64

PUTPN : IF _ERRPN<>0 THEN GOTO ERR
```

```
NREG=_PNPRN: AZZVARSYS 2

GETPN "20070103",NREG ,"","",0,0,0,"" : IF _ERRPN<>0 THEN GOTO ERR
 CONTO$=_PNCTO$(1)
 AZZVARSYS 2 '(altrimenti nel PG sarebbero presenti anche conti Iva)
          '_PNPAR$="07FR01/000003" (Partita Generale)
```

**PAGAMENTO PER PARTITA:**

_PNSCT$+_PNSPA$+_PNSDS$+_PNSID+_PNSTP$+_PNSIP (pagamento della 3^ rata della partita del conto in scadenza 30/04/07 con un pagamento in data 03/01/2007; eliminando _PNSDS$ si paga dalla rata più vecchia; eliminando anche _PNSIP si paga a saldo della primanota della partita -somma _PNIMP() del _PNCTO$(CONTO$)- a partire dalla rata piùvecchia)

-----

```
_PNSCT$(1)=CONTO$:_PNSPA$(1)="07FR01/000003":_PNSDS$(1)="20070430"
_PNSIP(1)=400.00
_PNCAU$="PG" : _PNDRE$="20070331" : _PNRPR$="" : _PNDDO$="20070103"
_PNCTO$(1)="601.00008" : _PNIMP(1)=  400
_PNCTO$(2)="201.00001" : _PNIMP(2)= -400   '(201.00001 è conto cassa)
-----
PAGAMENTO PER RIFERIMENTI DOCUMENTO:
_PNSCT$+_PNSCA$+_PNSSE+_PNSNU+_PNSDD$+_PNSDS$+_PNSID+_PNSTP$+_PNSIP
(paga 300 dalla rata più vecchia del doc FR1/8 del 14/01/06 del CONTO
-----
 _PNSCT$(1)=CONTO$: _PNSCA$(1)="FR": _PNSSE(1)=1: _PNSNU(1)=3:
 _PNSDD$(1)="20070103":_PNSIP(1)=300.00
 _PNCAU$="PG" : _PNDRE$="20070331" : _PNRPR$="" : _PNDDO$="20070103"
 _PNCTO$(1)="601.00008" : _PNIMP(1)=  300
 _PNCTO$(2)="201.00001" : _PNIMP(2)= -300   '(201.00001 è conto cassa)
-----
```

**PAGAMENTO PER DATA SCADENZA/TIPO PAGAMENTO:**

_PNSCT$+_PNSDS$+_PNSID+_PNSTP$

(paga tutte le rimesse "M" scadenti il 31/03/2007 del CONTO$ per un importo pari al saldo di primanota; ATTENZIONE l'esempio produce una registrazione non corretta nello scadenzario, perchè un pagamento non può essere superiore alla rata pagata)

-----

```
 _PNSCT$(1)=CONTO$ : _PNSDS$(1)="20070331" : _PNSTP$(1)="M"
 _PNCAU$="PG" : _PNDRE$="20070331" : _PNRPR$="" : _PNDDO$="20070103"
 _PNCTO$(1)="601.00008" : _PNIMP(1)=  500
 _PNCTO$(2)="201.00001" : _PNIMP(2)= -500   '(201.00001 è conto cassa)

PUTPN : IF _ERRPN<>0 THEN GOTO ERR
ERR: VIMSG 4, _ERRPN$
-----
RISULTATI:
-----
```

**RISULTATO DI PAGAMENTO PER PARTITA**

```
Data: 31/03/07 Valuta: EUR |ACCIAI srl A
Caus: PG   PAGAMENTO  |Saldo precedente..:    5.230,00A
Desc:      Co/ric:    |Imp. da scadenzare:         0,00
Numero pagamento....:        Del: 03/01/07|
001 Codice Descrizione conto Importo (EUR)|P a r t i t a           Saldo
601.00008 ACCIAI srl400,00D|07FR01/000001            430,00E
201.00001 CASSA400,00A|07FR01/000002          4.000,00E
  |07FR01/000003            800,00E

PARTITA: 07FR01/000003        Saldo:       800,00E
 Documento      del   Scadenza Tipo     Importo  Dt.Pag/Rif SaldoScadenza
 FR 1/     3 03/01/07 28/02/07 M         400,00E   400,00E
 FR 1/     3 03/01/07 31/03/07 M         400,00E   400,00E
 FR 1/     3 03/01/07 30/04/07 M         400,00E
 PG          03/01/07 30/04/07 MP       -400,00E  31/03/07

       Partita: 07FR01/000003
     ---------------------------
      Scadenza           Saldo
      28/02/07           400,00E
      31/03/07           400,00E
      30/04/07             0,00E
-----
```

**RISULTATO DI PAGAMENTO PER RIFERIMENTI DOCUMENTO**

```
Data: 31/03/07 Valuta: EUR  ACCIAI srl A
Caus: PG   PAGAMENTO   Saldo precedente..:    5.330,00A
Desc:      Co/ric:     Imp. da scadenzare:         0,00
Numero pagamento....:        Del: 03/01/07
001 Codice Descrizione conto Importo (EUR) P a r t i t a          Saldo
601.00008 ACCIAI srl300,00D 07FR01/000001            430,00E
201.00001 CASSA300,00A 07FR01/000002          4.000,00E
   07FR01/000003            900,00E

   PARTITA: 07FR01/000003        Saldo:       900,00E
 Documento      del    Scadenza Tipo     Importo  Dt.Pag/Rif SaldoScadenza
 FR 1/      3 03/01/07 28/02/07 M         400,00E
 PG           03/01/07 28/02/07 MP       -300,00E  31/03/07        100,00E
 FR 1/      3 03/01/07 31/03/07 M         400,00E   400,00E
 FR 1/      3 03/01/07 30/04/07 M         400,00E   400,00E

       Partita: 07FR01/000003
     ---------------------------
      Scadenza           Saldo
      28/02/07           100,00E
      31/03/07           400,00E
      30/04/07           400,00E
-----
```

**RISULTATO DI PAGAMENTO PER DATA SCADENZA/TIPO PAGAMENTO**

```
Data: 31/03/07 Valuta: EUR |ACCIAI srl A
Caus: PG   PAGAMENTO  |Saldo precedente..:    5.130,00A
Desc:      Co/ric:    |Imp. da scadenzare:         0,00
Numero pagamento....:        Del: 03/01/07|
001 Codice Descrizione conto Importo (EUR)|P a r t i t a          Saldo
601.00008 ACCIAI srl500,00D|07FR01/000001            430,00E
201.00001 CASSA500,00A|07FR01/000002          4.000,00E
  |07FR01/000003            800,00E
  |07PG03/01/07            -100,00E

   PARTITA: 07FR01/000003        Saldo:       800,00E
 Documento      del    Scadenza Tipo     Importo  Dt.Pag/Rif SaldoScadenza
 FR 1/      3 03/01/07 28/02/07 M         400,00E   400,00E
 FR 1/      3 03/01/07 31/03/07 M         400,00E
 PG           03/01/07 31/03/07 MP       -400,00E  31/03/07
 FR 1/      3 03/01/07 30/04/07 M         400,00E   400,00E

       Partita: 07FR01/000003
 (notare la registrazione non corretta      ---------------------------
  del pagamento del 03/01/07 di -100,00E)     Scadenza           Saldo
      28/02/07           400,00E
      31/03/07             0,00E
      30/04/07           400,00E
```

# _SINP$

In Collage contiene la stringa del campo input alla chiamata della routine, ad es. ON_ERR_CAMPO: (_SINP$ contiene il messaggio). Modificandola, contiene e inserisce nel campo il valore eventualmente modificato.

Esempio nella nota

ON_IN_MDNOTE: _SINP$="inserisco": END , ON_OUT_MDNOTE: _SINP$="": END

# _SC

variabili di struttura per PUTSC

| | |
|---|---|
| _SCCOD$() | Codice conto |
| _SCINIZ() | Saldo iniziale |
| _SCDARE() | Saldo dare |
| _SCAVER() | Saldo avere |
| _SCSALD() | Saldo dare/avere |
| _SCDATAAP$ | Data apertura |
| _SCDATACH$ | Data chiusura |
| _SCOPER | Operativita funzione PUTSC<br>0 = Scrittura record PN (default)<br>1 = Cancellazione record in data apertura _SCDATAAP$ |
| _SCNUMREG | Numero registrazioni PN eseguite |
| _ERRSC | Errore istruzione PUTSC se diverso da zero |
| _ERRSC$ | Descrizione errore _ERRSC |

Esempio IMOD-SPRIX per Saldi Contabili

```
'IMOD-SPRIX: apertura modulo
MODOPEN "S", "<c:\tmp\pdcsc1", "sc", "", 1, 131

'IMOD-SPRIX: definizione campi pagina
MODFIELD "", FROM 1 TO 1, FROM 2 TO 10 AS _MSCCOD
MODFIELD "", FROM 1 TO 1, FROM 16 TO 51 AS _MSCDESC$
MODFIELD "", FROM 1 TO 1, FROM 54 TO 64 AS _MSCINIZ NUM2
MODFIELD "", FROM 1 TO 1, FROM 67 TO 77 AS _MSCDARE NUM2
MODFIELD "", FROM 1 TO 1, FROM 80 TO 90 AS _MSCAVER NUM2
```

```
'IMOD-SPRIX: definizione sostituzioni
MODSFIELD "", _MSCCOD$, "CONTI", "S"

'IMOD-SPRIX: inizio ciclo di import dati
IMOD_CICLORIGHE:

  'IMOD-SPRIX: lettura file
  MODGET 'IMOD-SPRIX

  'IMOD-SPRIX: chiamata MODPUT per scrittura dati
  IF _ERRMOD = 0 AND _MODSCRIVI = 1 THEN MODPUT "sc" 'IMOD-SPRIX

  REM --- IMOD:controllo errore e uscita
  IF _ERRMOD <> 0 THEN _MODTOTRIG = 0 'IMOD-SPRIX
IF _MODNUMRIG < _MODTOTRIG THEN GOTO IMOD_CICLORIGHE 'IMOD-SPRIX

'IMOD-SPRIX: chiusura modulo
 MODCLOSE
'IMOD-SPRIX: messaggi di elaborazione
 MODVIMSG  'IMOD-SPRIX
'IMOD-SPRIX: messaggio errore
 IF _ERRMOD <> 0 THEN VIMSG 5, _ERRMOD$ 'IMOD-SPRIX
```

## _SCORRI

In Sprix/Collage condiziona le successive INPUT; se _SCORRI=0 sono interpretate come delle INPUT, altrimenti (se_SCORRI=1) come delle PRINT con: azzeramento di _TF, la sola visualizzazione di <str$> della INPUT, senza richiesta di immissione, senza variazioni di posizione della funzione o cambiamenti di colore del cursore.

## _STPMMRIG

Variabile collage utilizzata per le stampe personalizzate "Documenti di Magazzino", "Movimenti di Magazzino".Riporta il numero di riga elaborata del documento elaborato.

Esempio Collage

Apre una finestra che riporta la descrizione del campo di INPUT in cui si trova il cursore (compilazione in menu AUTO)

```
ON_START:
 FLAG=1
END

ON_IN_ALL:
 IF FLAG=1 THEN GOTO SALTO1
 GOSUB FINESTRA
 SALTO1:
END

ON_OUT_ALL:
 IF FLAG=1 THEN GOTO SALTO2
 CHIUDIFIN NUMFIN
 SALTO2:
 FLAG=0
END

FINESTRA:
 NUMFIN=AUTOFIN() : X=GETPOSCURSX(2) : Y=GETPOSCURSY(2)
 L=LEN(_NCAM$)
 IF Y=20 THEN Y=18
 APRIFIN NUMFIN,11,_NFIN,Y+1,X,1,10+L
 LOCATE 1,1:COLOR 3: PRINT "Cod.Camp.:"+_NCAM$;
RETURN
```

# _STPTNO

Variabile collage relativa alle stampe personalizzate Movimenti di magazzino Documenti di magazzino. Impostata ad "1" esclude la riga dalla successiva stampa. Saranno escluse dalla stampa tutte le righe finche la variabile non viene nuovamente impostata a 0(zero).

# _TF

In Sprix/Collage ritorna il codice numerico (dichiarato con SETTF prima del-la INPUT) del tasto utilizzato per uscire dal campo di INPUT o con l'evento ON_OUT_?. Assume valore 1=Invio (anzichè 501), 1=Freccia su, -1=SH+Tab e -1=Freccia giù in assenza di SETTF e in evento ON_OUT_? (in cui vale 1000 se cliccato il mouse). _TF è globale e modificabile: se cambiato il contenuto, viene considerato come tasto utilizzato per l'uscita dal campo in sostituzione di quello premuto dall'utente (esempio, annullamento pressione del tasto funzione F8: ON_OUT_MDQTA: IF _TF=608 THEN _TF=0 : END). Alla chiamata della routine collage _TF è automaticamente settata, se non è attivata la relativa etichetta _TF non è valorizzata.

# _TP

variabili di struttura archivi Distinta Base Primaria (GET-PUT-DEL)

In rosso evidenziati i campi minimi obbligatori

| TESTATA | |
|---|---|
| _TPCOD$ | (cod.articolo con distinta base) |
| _TPSVI$ | (articolo per calcolo costo) |
| _TPCTR$ | (controllo esistenza/disponibilità) |
| _TPPFP$ | (prodotto finito padre di default) |
| _TPAGG$ | (tabella colori di aggancio) |
| _TPSML$ | (categoria semilavorato) |
| _TPUSA$ | (usa DBVC/DBVO) |
| _TPCOL$ | (gestione collegamento PF->SML) |
| _TPUMP$ | (tipo unità di misura utilizzato (P o S) |
| _TPMMP | (magazzino materie prime di default) |
| _TPDEC | (numero di decimali) |
| _TPCLI$ | (cod. cliente di default) |
| _TPOAP | (ore di approntamento) |
| _TPORE | (ore per lotto) |
| _TPPNR | (lotto di produzione) |
| _TPKOE | (coefficente) |
| _TPKOC(..,16) | (consumi per taglia personali) |
| _TPKOCXL(..,16) | (consumi per taglia personali (Tg 17-32) |
| **FASE** | |
| _TPDES$ | (descrizione della fase) |
| _TPFSC$ | (tipo condizione della fase) |
| _TPULT$ | (ulteriore descrizione della fase) |
| _TPNQT$ | (quantità per riferimento) |
| _TPDOC$ | (documenti in stampa) |
| _TPNOT$() | (gruppo di 10 note della fase) |
| **COMPONENTE** | |
| _TPIND$() | (indice) |
| _TPART$() | (cod. componente) |
| _TPQTA$() | (quantità) |
| _TPCOE() | (coefficiente di conversione) |
| _TPEVI$() | (evidenza o fase di disimpegno rapido) |
| _TPNEC$() | (negazione cond. cliente) |
| _TPNED$() | (negazione cond. data) |
| _TPNET$() | (negazione cond. taglie) |
| _TPSSS$(,n) | (array art. sostitutivi -tutti e 4-) |
| _TPMAG() | (magazzino componente -in standby-) |
| _TPNEG$() | (negazione condizioni) |
| _TPCON$() | (tipo condizione "A" o "S")*a |
| _TPCAR$() | (condizione (cod.articolo) |
| _TPUNM$() | (unità misura utilizzata: 1 -prim- o 2 -second-) |
| _TPCCL$() | (cod. cliente condizione) |
| _TPCDD$() | (da data) |
| _TPCDA$() | (a data) |
| _TPCNOn$() | (nota del componente; 'n' assume un valore da 0 a 9) |
| _TPTAG$() | (condizione per taglia) |
| _TPCTS$() | (tipo sostituzione componente) |
| _TPSOS$() | (articolo in sostituzione) |
| _TPDIR$() | (direzione P=precedente S=successiva) |
| _TPLOT$() | (spezza lotto; se "" -> viene azzerato (errato)) |
| _TPEDE$() | (escludi da deposito S/N) -GET/PUT- |
| _TPESC$() | (escludi da scarico S/N) -GET/PUT- |
| _TPEST$() | (lavorazione esterna S/N) -GET/PUT- |
| _TPCST$() | (tipo di costo - ,U,S,M,V,P-) -GET/PUT- |
| _TPCOS() | (costo lavorazione) -GET/PUT- |
| _TPEFO$() | (codice Fornitore)-GET/PUT- |

*Note:*

*a - _TPCON$() accettava anche valori "C","T","D" e pertanto, se tali valori sono utilizzati nel sorgente in essere, questi dati sono ignorati dalla istruzione (es.la PUTDB non va in errore ma ignora); se presente una IF che testa queste valorizzazioni occorre modificare la IF, altrimenti la condizione viene ignorata.

Esempio: Dato l'articolo ARTDBP con Fase 1 a due Componenti (A e B) e Fase 2 a due Componenti (C e D), legge la DBP e ne cancella l'ultima Fase per ricrearla con due altri componenti (E e F).

```
FOR FAS=1 TO _NFASI
    GETDB "ARTDBP",FAS : IF _ERRDB<>0 THEN VIMSG 4,_ERRDB$
```

```
        FOR IND=1 TO _NCOMP
                PRINT "Fase "+STR$(FAS,0,0)+" Compo. "+STR$(IND,0,0)+" :"+_TPART$(IND)
        NEXT
NEXT
VIMSG 1, "Cancella e sostituisce l'ultima Fase con componenti E - F"
DELDB "ARTDBP",_NFASI : IF _ERRDB<>0 THEN VIMSG 5, _ERRDB$
 _TPDES$="NUOVA FASE":_TPULT$="ULT. DESCRIZ.":_TPDOC$="C": _NCOMP=2
 _TPIND$(1)="5":_TPART$(1)="COMP_E":_TPQTA$(1)="5":_TPUNM$(1)="1"
 _TPIND$(2)="6":_TPART$(2)="COMP_F":_TPQTA$(2)="5":_TPUNM$(2)="1"
 PUTDB _NFASI : IF _ERRDB<>0 THEN VIMSG 4,_ERRDB$ : END
```

Esempio: sviluppo ARTDBP per cliente "201.00001", q.ta=10*3,5 (il componente di DBP con condizione cliente viene sviluppato, diversamente no)

```
SVILDB "ARTDBP","","201.00001",0,10,3.5: IF _NSVILDB=0 THEN GOTO ERR1
 FOR I=1 TO _NSVILDB
 IF osval(I)=0 THEN GOTO PROSSIMA1
 PRINT "  Fase: "+STR$(osfas(I),0,0)+" , Comp.: "+osart(I);:
 PRINT " , Q.tà: "+STR$(osqta(I),0,0)
 '----> eventuale righe di calcolo
 PROSSIMA1: NEXT
 PRINT ""
 VIMSG 1, "termine sviluppo DBP"
 ERR1: IF _NSVILDB=0 THEN VIMSG 1, _ERRSVILDB$
```

sviluppo ARTDBPTAG (a Taglie) per cliente "201.00001" con aggancio a DBVO della riga ordine [ocnri()] per una q.tà di 75 (10 per la 1^ Tg. 20 per la2^ Tg. e 45 per la 4^ Tg.). L'OC su cui è inserita una 'DBP a taglie (che preveda la possibilità di DBV) deve ricevere la modifica di un componente.

Lo sviluppo moltiplica le quantità per quella di prodotto finito su riga.

```
SVILDB "ARTDBPTAG","","201.00001",631,0,10,20,0,45
IF _NSVILDB=0 THEN GOTO ERR2
FOR I=1 TO _NSVILDB
IF osval(I)=0 THEN GOTO PROSSIMA2
PRINT "  Fase: "+STR$(osfas(I),0,0)+" , Comp.: "+osart(I);:
PRINT " , Q.tà: "+STR$(osqta(I),0,0)
 '----> eventuale righe di calcolo
PROSSIMA2: NEXT
PRINT ""
VIMSG 1, "termine sviluppo DBVO su articolo con DBP a taglie"
ERR2: IF _NSVILDB=0 THEN VIMSG 1, _ERRSVILDB$
```

# _TPLOT

_TPLOT? variabili di struttura archivi Anagrafica Tipo Lotto (GET-PUT-DEL)

| | |
|---|---|
| _TPLOTCOD$ | (Codice tipo lotto; "**" se nuovo inserimento) |
| _TPLOTDESC$ | (Descrizione tipo lotto) |
| _TPLOTSTAT$ | (Stato tipo lotto) |
| _TPLOTCUNI$ | (Chiave utente univoca(S/N) |
| **PARAMETRI LOTTO** | |
| _TPLOTAPGS$ | (Gestione: E/D/e/d) |
| _TPLOTAPTP$ | (Tipo F/L/C/D) |
| _TPLOTAPCO$ | (Mantenere colli quantita S/N) |
| _TPLOTAPCL$ | (Mantenere colli peso lordo S/N) |
| _TPLOTAPSL$ | (Spezza quantita lotto S/N) |
| _TPLOTAPQR$ | (Quantita residua lotto N/A/C) |
| _TPLOTSCAQ$ | (Avviso quantità insuff.: N/E/D) |
| _TPLOTCATP$ | (Tipo controllo M/F/D/A) |
| **CAMPI PERSONALIZZATI** | |
| _TPLOTDUCD$(999) | (Codice tipo lotto - se nuovo, i primi 2 car. sono "**" - |
| _TPLOTDUET$(999) | (Etichetta) |
| _TPLOTDUTP$(999) | (Tipo) |
| _TPLOTDUDI(999) | (Dimensione) |
| _TPLOTDUOB$(999) | (Obbligatorio S/N) |
| _TPLOTDUOR(999) | (Riga ordine visualizzazione) |
| _TPLOTDUVL$(999) | (Valore iniziale default -anche nome file-) |
| _TPLOTDUCH(999) | (num elem. cod. utente (chiave) (max 63 ) |

Esempio: scrive tipo lotto generico, lo duplica personalizzandolo e infine cancella l'originario tipo lotto

```
_TPLOTCOD$="**" : _TPLOTDESC$="Descriz. tipo lotto"
 _TPLOTSTAT$="C" : _TPLOTCUNI$="S"
 PUTTPLOT : IF _ERRTPLOT<>0 THEN VIMSG 4, _ERRTPLOT$
 VIMSG 1, "Ha scritto il tipo lotto generico|"+_TPLOTCOD$
```

```
TPL$=_TPLOTCOD$
 AZZVARSYS 18 : GETTPLOT TPL$,2 : IF _ERRTPLOT<>0 THEN VIMSG 4, _ERRTPLOT$
_TPLOTCOD$="**" : _TPLOTDESC$="Descriz. NUOVO tipo lotto"
 _TPLOTSTAT$="C" : _TPLOTCUNI$="S"
 _TPLOTAPGS$="E" : _TPLOTAPTP$="F" : _TPLOTAPCO$="S" : _TPLOTAPCL$="S"
 _TPLOTAPSL$="N" : _TPLOTAPQR$="N" : _TPLOTSCAQ$="N" : _TPLOTCATP$="M"
_TPLOTDUCD$(1)="** 3": _TPLOTDUET$(1)="Etichetta" : _TPLOTDUTP$(1)="A"
 _TPLOTDUDI(1)=20 : _TPLOTDUOB$(1)="S" : _TPLOTDUOR(1)=1
 _TPLOTDUVL$(1)="Valore iniziale": _TPLOTDUCH(1)=4
PUTTPLOT : IF _ERRTPLOT<>0 THEN VIMSG 4, _ERRTPLOT$
 VIMSG 1, "Ha duplicato il tipo lotto generico| e lo ha personalizzato"
DELTPLOT TPL$ : IF _ERRTPLOT<>0 THEN VIMSG 4, _ERRTPLOT$
 VIMSG 1, "Ha cancellato l'iniziale tipo lotto generico"
```

# ABILCOLLAUTO

### ABILCOLLAUTO <nomefile$>,<abil>

In Collage abilita (<abil>=1, default) o disabilita (<abil>=0) il caricamento del nome file Collage automatico senza estensione <nomefile$>, per impedirne l'eventuale ricaricamento quando fermato con STOPCOLLAGE.

# ABS

### ABS(<num>)

In Sprix/Collage ritorna il valore assoluto di <num>.

Esempio
dati A=-1234: B=4 il risultato di ABS((A*B)/2) o ABS(-1234*2) è 2468

# ACAPO

### ACAPO$()

In Sprix/Collage ritorna una stringa di caratteri da inviare alla stampante (con formato testo) attualmente aperta per effettuare un salto di riga (per andare a nuova riga). Se nessuna stampante è aperta, ritorna una stringa  vuota. Es. LPRINT ACAPO$();

# ADCOMMERCIALE

### ADCOMMERCIALE ModAgg$,MM$,AnnoInizElab, AnnoFineElab,PrzCalcMarg$,OC$, OF$, MA$

| | |
|---|---|
| ModAgg$ | (Modalità aggiornamento). Valori accettati "C" = completa o "V" = solo variati |
| MM$ | Movimenti di magazzino). "S" = si o "N" = no |
| AnnoInizElab | Anno inizio elaborazione |
| AnnoFineElab | Anno fine elaborazione |
| PrzCalcMarg$ | (Prezzo calcolo margine). Valori accettati "M" = movimento o "A" = anagrafica |
| OC$ | (Ordini clienti) . "S" = si o "N" = no |
| OF$ | (Ordini fornitori) . "S" = si o "N" = no |
| MA$ | (Matrici) . "S" = si o "N" = no |

In Sprix esporta i dati commerciali del gestionale nel database di AD. Nel caso in cui MM$ = "N" le variabili degli anni possono essere vuote. Ritorna eventuali errori nelle variabili **_ERRADCOM, _ERRADCOM$.**

# ADPCOMMERCIALE

### ADPCOMMERCIALE <annorif>,<annoprec>,<esplostrut>

In Sprix esporta i dati commerciali del gestionale nel database di ADP (funzione obsoleta già integrata nel gestionale); gli anni da confrontare sono <annorif> e <annoprec> dove il primo è > del secondo; <esplostrut> esplode gli articoli strutturati (0=no, 1=si). In passato era prevista anche un'esportazione ADPINDICI per le analisi di bilancio.

# APPENDISPX

### #APPENDISPX <nomefile$1> [<nomefile$2> <nomefile$3>...]

Nella compilazione di Sprix/Collage aggiunge il programma <nomefile$1> in coda al sorgente scritto; il primo parametro <nomefile$1> è obbligatorio e va inserito senza estensione fra i simboli <..>, gli altri parametri (fra [..]) sono facoltativi. La collocazione di #APPENDISPX (anche quando sono varie e su più righe) va assolutamente in testa al programma e serve per agganciare altri programmi al codice in uso, per renderlo più snello e non ripetitivo; la collocazione ottimale di questi programmi è nella voce Libreria (non compilabile e non eseguibile autonomamente) per essere riutilizzabili/disponibili per più

Sprix/Collage. Sprix è un linguaggio procedurale elaborato riga per riga, la cui sequenza di routine da eseguire è pilotabile attraverso la dichiarazione di ETICHETTE (label) di massimo 29 caratteri maiuscoli terminati da ':' e non duplicabili (es. INIZIO:, RIPETI: FINE:,...) che servono per definire le porzioni del programma (routine) da chiudere opportunamente, specialmente in Collage, con istruzione END. Collocare il codice in coda al sorgente non è un problema: le routine di libreria devono iniziare con etichetta eseguibile con GOSUB del programma chiamante per terminare con RETURN che restituisce il controllo al medesimo programma. Per la stessa ragione è consigliato terminare lo Sprix chiamante con END.

# APRIAZIE

### APRIAZIE <sigla$>,<data$>

In Sprix chiude l'azienda attualmente in uso e apre l'azienda <sigla$> in data <data$> (cambia l'archivio e ridisegna la videata uscendo della routine; ciclabile solo nella tabella 'sà). Esempio APRIAZIE "SIG","20060322".

Esempio: per gli anni delle aziende installate (eccetto l'azienda A74) cicla e stampa alcuni dati delle tabelle 'pc', 'pn', 'ar', 'mm'

```
|AZORIG$=azsig:DATORIG$=sxdat
|X=0:DIM AZ$():DIM PRIM():DIM ULT():DIM ANN():DIM ANNI(): DIM
NUM()
 N|**************** AZIENDE INSTALLATE **********************
     |+-----------------------------------------
 Cod.Tab.:sa  DATI GENERALI - AZIENDE INSTALLATE
|Ordine 3:sasig()
     |+-----------------------------------------
 sa   |X=X+1:AZ$(X)=sasig():
PRIM(X)=sapri():ULT(X)=sault():NUM(X)=sanum()
 sa  N|
 sa  N|~X\-~sa-~sap-~sau-~saest()\\\\\\\\\\\-~sa-~-~sa - ANNI =
~NU
     |
 FOR I=1 TO X: IF AZ$(I)="A74" THEN GOTO SALTA: 'SLEEP(1000)
|FOR J=NUM(I)-1 TO 0 STEP -1
IF NUM(I)>1 THEN ANNO=PRIM(I)+J ELSE ANNO=PRIM(I)
     |
APRIAZIE AZ$(I),STR$(ANNO,0,0)+"1225"
IF _ERRAZIE<>0 THEN VIMSG 4, _ERRAZIE$
PRINT"ANNO GESTITO "+STR$(ANNO,0,0)+ " " +AZ$(I):PRINT
STRING$(20," ")
N|
N|PIANOCONTI ~AZ ANNO ~ANN
N|
     |+-----------------------------------------
Cod.Tab.:pc  PIANO DEI CONTI - PIANO DEI C. PC/CLI/FORN
     |+-----------------------------------------
pc  N|           ~AZ-~pccod()\
N|
N|PRIMANOTA  ~AZ ANNO ~ANN
N|
     |+-----------------------------------------
Cod.Tab.:pn  CONTABILITÀ - PRIMA NOTA INTESTAZ.
Ordine 3:pnprn()+pndre()
     |+-----------------------------------------
pnI N|           ~"prn\"+STR$(ANNO,-~pnprn()\\-~pndre()\\
 N|
N|      ARTICOLI ~AZ ANNO ~ANN
N|
     |+-----------------------------------------
 Cod.Tab.:ar  MAGAZZINO - ANAGRAFICA ARTICOLO
     |+-----------------------------------------
ar  N|~"articoli\"-~arcod()\\\\\\\\\\\
N|
N|     MOVIMENTI ~AZ ANNO ~ANN
N|
     |+-----------------------------------------
Cod.Tab.:mm  MOVIMENTI DI MAGAZZINO - TESTATA DOCUMENTO
Ordine 3:mmsig()+mmser()+mmnum()+mmcli()
```

```
          |+-----------------------------------------
mmI N|~"movimenti\"+STR$(ANN-~m-~m-~mmnum
 SALTA:NEXT:NEXT
APRIAZIE AZORIG$,DATORIG$: IF _ERRAZIE<>0 THEN VIMSG 4, _ERRAZIE$
END
          |
```

## APRIAZIE_EXT

### APRIAZIE_EXT <sigla$>,<data$>,<codSottAz$>

Nella gestione di sottoaziende (es.attività Iva) assume la diversa sintassi APRIAZIE_EXT <sigla$>,<data$>,<codSottAz$> per aggiungere in <codSottAz$> il numero di codice sotto azienda gestito: da 1 a 256 (30 se attività Iva) oppure 'G' (Generale) o 'P' (Prevalente, cioè default se parametro = ""). La 'G' (per convenzione è la numero -sxcsa- 651) può ciclare solo i suoi datie non può leggere in modalità 'T' (primenote delle sotto attività come avviene in revisione primanota) salvo singoli APRIAZIE_EXT su ciascuna sotto azienda. Vedi APRIAZIE.

Esempio APRIAZIE_EXT "A36",sxdat,"1".

## APRIFIN

### APRIFIN <nfin>,<colore>,<rifnfin>,<rig>,<col>,<alt>,<larg>

| | |
|---|---|
| nfin | Numero delle finestra. Valori accettati da 1 a 9 |
| colore | Colore finestra.<br>1 e 2 normale,<br>3 aiuto,<br>4 messaggio,<br>5 errore,<br>9 base<br>e le stesse, ma senza il bordo 11, 12, 13, 14, 15 e 19 |
| rifnfin | Riferimento di apertura della finestra rispetto la precedente<br>da 0 base del gestionale fino a 9; in Collage è in negativo per indicare quelle del gestionale. |
| rig | Da 1 al numero massimo di righe gestite dal desk. Se < 0 si intende in alto rispetto <rifnfin>,se = 0 si intende la riga centrale |
| col | Da 1 al numero massimo di colonne gestite dal desk. se < 0 si intende a sinistra rispetto <rifnfin>, se = 0 si intende la colonna centrale e se anche <rig>= 0 la finestra è centrata nell'area di lavoro. |
| alt | Da 1 a massimo 21 |
| larg | Da 1 a massimo 118 |

La dimensione massima delle righe e delle colonne è dipendente dal tipo di video che si sta utilizzando.

*Nota Bene: La gestione delle finestre di passepartout è di 19 righe per 94 colonne.*

In Sprix/Collage apre una finestra numero <nfin> (da 1 a 9 anche contemporaneamente + quella n.0 base del gestionale, necessariamente aperta in assenza di alcuna apertura), del tipo <colore> (colore se consentito da interfaccia;1 e 2 normale, 3 aiuto, 4 messaggio, 5 errore, 9 base e le stesse, ma senza il bordo 11, 12, 13, 14, 15 e 19), riferita al numero finestra <rifnfin> (da 0 base del gestionale fino a 9; in Collage è in negativo per indicare quelle del gestionale) a cui si riferiscono le coordinate di posizionamento dello schermo <rig> (riga da 1 a numero massimo di righe del desk; se < 0 si intende in alto rispetto <rifnfin>,se = 0 si intende la riga centrale) e <col> (colonna da 1 al numero massimo di colonne del desk; se < 0 si intende a sinistra rispetto <rifnfin>, se = 0 si intende la colonna centrale e se anche <rig>= 0 la finestra è centrata), di altezza <alt> (1-22 righe) e larghezza <larg> (1-118 colonne).

Es. APRIFIN 1,1,0,5,10,10,30

Attiva la finestra aperta (e disattiva le precedenti) con posizione cursore in riga 1 e colonna 1. <nfin> è indicativo (non ne indica la sequenza, poichè Sprix considera la sequenza effettiva con cui le finestre vengono aperte, anche con numeri a caso. Le righe 1,2,3,24,25 sono riservate:

## APRIFIN_TIT

### APRIFIN_TIT <nfin>,<colore>,<rifnfin>,<rig>,<col>,<alt>,<larg>,<tit>,<tasti>

| | |
|---|---|
| nfin | Numero delle finestra. Valori accettati da 1 a 9 |
| colore | Colore finestra.<br>1 e 2 normale,<br>3 aiuto,<br>4 messaggio,<br>5 errore,<br>9 base<br>e le stesse, ma senza il bordo 11, 12, 13, 14, 15 e 19 |

| rifnfin | Riferimento di apertura della finestra rispetto la precedente<br>da 0 base del gestionale fino a 9; in Collage è in negativo per indicare quelle del gestionale. |
|---|---|
| rig | Da 1 al numero massimo di righe gestite dal desk. Se < 0 si intende in alto rispetto <rifnfin>,se = 0 si intende la riga centrale |
| col | Da 1 al numero massimo di colonne gestite dal desk. se < 0 si intende a sinistra rispetto <rifnfin>, se = 0 si intende la colonna centrale e se anche <rig>= 0 la finestra è centrata nell'area di lavoro. |
| alt | Da 1 a massimo 21 |
| larg | Da 1 a massimo 118 |
| tit | Titolo delle finestra |
| tasti | Codice del tasto che si vuole visualizzare sulla parte destra del video. Vedi SETTF |

La dimensione massima delle righe e delle colonne è dipendente dal tipo di video che si sta utilizzando.
In caso di video 4:3 il massimo dell'area video disponibile è 21 per 98 colonne
In caso di video 16:9 il massimo dell'area video disponibile è 21 righe per 118 colonne

L'estensione APRIFIN_TIT prevede due ulteriori parametri: la stringa <tit> per dare un titolo alla finestra e la stringa <tasti> per indicare i tasti attivi con descrizione (vedi SETTF); quest'ultimo caso è applicabile solo in Sprix, con visualizzazione del tasto con descrizione nel elenco verticale di pulsanti sulla destra del gestionale (anzichè nell'ultima riga della finestra aperta come invece farebbe SETTF) ed attivazione attraverso la SETTF.

```
Esempio Sprix:
NF=AUTOFIN()
APRIFIN_TIT NF,1,0,3,3,10,10,"Titolo","502Descrizione"
SETTF "502Descrizione"
INPUT "N",N,1
```

# APRISTAMP

### APRISTAMP <numstamp>,<nomemod$>

In Sprix/Collage apre la stampante numero <numstamp> (da 1 a 126 per quante installate; 0=soppressa, 127=video) e con richiesta nome modulo <nomemod$> da inserire, es. "SCHEDE CONTABILI" (se "" non è visualizzato il messaggio di richiesta di inserimento del modulo). I <numstamp> sono memorizzati anche nel file di configurazione (interrogabile).

```
Esempio:
APRISTAMP 127,""
LPRINT "1234567890"
CHIUDISTAMP
```

Dopo l'utilizzo della stampante, questa va chiusa e liberata per altri terminali con la CHIUDISTAMP, che chiude la stampante attualmente aperta (anche automaticamente da altre operazioni di stampa).

APRISTAMP e CHIUDISTAMP valorizzano _NUMSTAMP che identifica la stampante attualmente aperta coi seguenti valori: da 1 a 126 (numero di stampante installata e attualmente aperta), 0 (stampa soppressa), 127 (stampa a video), -1 (stampa annullata incondizionatamente da utente con [Esc/Fine] al messaggio d'inserimento modulo o con CHIUDISTAMP); _NUMSTAMP non identifica il numero di stampante selezionata dal utente nella stampa del gestionale.

All'evento di stampa e/o termine di stampa, la stampante definita nella configurazione non è bypassabile dall'apertura di una stampante diversa (e con il messaggio "stampante occupata o già in uso"): attendere il termine dalla stampa (e/o richiamare APRISTAMP con Sprix) oppure intercettare l'evento successivo con la gestione di un flag di attivazione (per evitare la ripartenza della stampa). I formati di stampa (o automatismi dipendono dalla configurazione della stampante (esempio, caratteri "!" per l'email).

CHIUDISTAMP può rilasciare il messaggio "immettere uno dei seguenti valori (S/N), ripetere l'immissione", quando si esce con ESC dalla routine Collage che, all'ON_612_? su campo non vuoto esegue APRISTAMP (ad esempio 127) con LPRINT seguita dalla CHIUDISTAMP: normalizzare azzerando con AZZTAST.

# APRISTAMP_EMAIL

### APRISTAMP_EMAIL <numstamp>,<mittente$>,<destinatario$>,<cc$>,<ccn$>,<oggetto$>,<testo$>,<firma$>,<nomeallegato$>,<allegato$>,[""]

| numstamp | numero della stampante che deve essere obbligatoriamente di tipo "E" |
|---|---|
| Mittente$ | Indirizzo email del mittente |
| Destinatario$ | Indirizzo email del destinatario |
| cc$ | Indirizzo email del copia conoscenza |
| ccn$ | Indirizzo email del copia conoscenza nascosto |
| oggetto$ | Oggetto della mail |
| testo$ | Testo da inserire prime della stampa della mail. |

| firma$ | Firma della mail |
|---|---|
| nomeallegato$ | Nome del file allegato |
| allegato$ | Percorso del file da allegare alla mail |

La funzione ritorna la variable di errore _ERRSTPEMAIL se la stampante non è di tipo Email. I parametri della funzione rispettano fedelmente quelli che sono i campi della stampante Email del gestionale. Per le specifiche di ogni singolo campo si rimanda al manuale.

## APRISTAMP_EXT

### APRISTAMP_EXT <numstamp>,<nomemod$>,<lun>

Utilizza gli stessi parametri dell'apristamp con l'aggiunta del parametro <lun> che consente di stabilire la lunghezza in caratteri della riga di stampa (n. colonne): se lun=0 è impostata a 220, se lun<79 è impostata a 79, se lun>220 è impostata a 220.

Esempio:
```
APRISTAMP_EXT 127,"",79
LPRINT "1234567890"
CHIUDISTAMP
```

## ARRO

### ARRO(numero,tipo,arr)

| numero | numero intero da arrotondare |
|---|---|
| tipo | tipo di arrotondamento 1 = difetto, 2 = medio-difetto, 3 = medio-eccesso, 4 = eccesso |
| arr | valore dell'arrotondamento |

In Sprix/Collage ritorna il <numero> intero arrotondato in base al valore intero di arrotondamento <arr> (al centesimo di Euro è 0.01) e in base al tipo <tipo> di arrotondamento (1 = difetto, 2 = medio-difetto, 3 = medio-eccesso, 4 = eccesso).

Esempio
dato N=12500: ARRO(N,2,1000) vale 12000, ARRO(N,3,1000) vale 13000
In caso di numero con decimali (es N=12,536 da arrotondare con tipo 1 al centesimo) può adottarsi la forma: B=ARRO (N*100,1,1)/100

## ARTDBP

### ARTDBP(<codart$>)

In Sprix/Collage controlla se l'articolo con codice principale <codart$> è di produzione (ritorna 1) oppure no (ritorna 0).

## ASC

### ASC(<str$>)

In Sprix/Collage ritorna il valore numerico ASCII del primo carattere della stringa <str$>.

Esempio.
A$="A1B" il risultato di ASC(A$) è ASC("A") cioè 65

## ASCMIRROR

### ASCMIRROR <tabella$>,<dir$>,<file$>

| tabella | Tabella da elaborare |
|---|---|
| dir$ | Directory dove salvare il file$ |
| file$ | Nome del file da generare |

Effettua la lettura di tutti i dati contenute nei record del nome di <tabella$> del dizionario (elencato nella directory del disco della versione"gest\doc\conces\dbmirror\"), esportando in sovrascrittura/precancellazione del supporto queste informazioni su file ASCII(ASCMIRROR)

Richiede il nome <file$> (minuscolo se Linux; se "" il default è il nome di<tabella$.ASC>) e il relativo percorso <dir$> (se "" il default è directory dati azienda); l'errore di accesso al file è rilasciato da _ERRFILE$. Il file è in formato CSV se utilizzato il ';' come separatore fra le colonne. Le stringhe non sono delimitate da alcun carattere, se nell'informazione di un campo stringa ci fosse almeno un ';' potrebbe comportare discrepanza tra colonne e dati. La prima riga è riservata ai nomi delle voci del dizionario per il battesimo delle colonne. I campi data non valorizzati nel gestionale sono convertiti in NULL (ASCMIRROR)

Viene visualizzato lo stato di avanzamento progressivo dei record trattati e si può interrompere con tasto [Esc] (previa impostazione di _INTER) con riempimento del file/database fino a quanto eseguito (l'interruzione dell'elaborazione della tabella corrente non interrompe gli eventuali mirroring successivi). Non è eseguibile in ciclotabellare, di cui valgono alcune regole, come LEGGIDESC per l'articolo descrittivo nel documento, l'elaborazione annuale per anno di ingresso in azienda;

> *Attenzione: il mirroring (anche quello in tempo reale) dei documenti di anni diversi con chiave identica (ma che non comprenda l'anno, es. sigla+serie+numero+conto a numeratore unico e non annuale) si sovrascrivono.*

Il mirroring di interi archivi impegna il sistema, se è già appesantito da altre condizioni (dbmirror in tempo reale, reti sovraccariche, computer non performanti..) è preferibile l'export delle tabelle necessarie ai soli dati di interesse da esportare con procedure batch di fine lavoro oppure interrogazioni di database replicati in chiaro o infine mirroring ottimizzati.

Il mirroring non va eseguito nel ciclotabellare. La ASCMIRROR valorizza la _ERRASCMIR che rilascia il codice dell'esito della operazione effettuata:

| 0 | esito positivo dell'operazione (nessun messaggio, il campo è vuoto); |
|---|---|
| 5 | tabella non gestita; |
| 6 | tabella non ancora implementata; |
| 7 | archivio sprix non trovato; |
| 8 | tipo dato DB diverso da tipo dato sprix; |
| 9 | campo sprix non trovato; |
| 10 | dimensione del campo sprix superiore alla dimensione del campo DB; |
| 11 | errore interno sul file di risorsa; |
| 12 | directory di destinazione del file di output non esistente; |

# ASCMIRROR_EXT

### ASCMIRROR_EXT <tip>,<rif$>,<dir$>,<file$>,<slot>,<att>,<var>,<azz>,[vis]

Ottimizza la ASCMIRROR sulle tabelle Anagrafiche Piano dei Conti e Articoli, riducendo il volume dei dati esportato dalle tabelle:

PC_ PP_ PA_ e AR_ AH_ AF_ AX_ AY_ AL_ AB_ AW_ AQ_ AO_ AE_.

La riduzione avviene con i flag <slot>,<att>,<var>,<azz>. Nel menu Aziende -Parametri di base - tabella Stato anagrafiche del Gestionale, si possono stabilire fino a 8 tipi di slot di esportazione (da utilizzare in altrettante applicazioni Sprix (esempio E-COMMERCE) per le quali definire, con la pressione del tasto [SHF11] e la selezione di 'Stato anagrafica' sul conto o sul articolo nelle rispettive anagrafiche, lo stato ATTIVO oppure NON ATTIVO dello<slot> sulla variazione del dato: il singolo record è quindi esportato o meno in base a quanto stabilito con lo slot>sul flag di variazione del singolo record (che viene alzato dal Gestionale alla modifica del record). Nella operazione PUT? la scrittura default è N per 'ATTIVÒ e S per il 'VARIATÒ;gli elementi 9-12 sono invece di sola lettura.

Gli <slot> della DBMIRROR e ASCMIRROR sono in comune: una esportazione su database con DBMIRROR determina la non esportazione su file dell'ASCMIRROR sul medesimo slot.

Lo <slot> è quindi un numero da 1 a 8; se = 0 non gestisce questa ottimizzazione attraverso il flag numerico <var> per l'export dei soli record variati dall'ultimo aggiornamento (se <var>=1) o di tutti i record con cancellazione (se <var>=0) o drop (se <var>=2) della tabella nel database (rendendo quindi irrilevante il flag 'Variatò dello <slot>).

Il <var> dipende comunque dal flag numerico <att> (se <att>=0 tratta tutti i record, se <att>=1 solo se sono attivi). Terminata l'esportazione, per evitare successivi ri-trattamenti del record, il <var> di tutti gli slot (relativi al intero archivio anagrafico e non della singola tabella) viene azzerato se <azz>=1 (se <azz>=0 non azzera <var>), poichè se il flag numerico <tip> è = 0 si stabilisce che il riferimento <rif$> è il nome di una tabella appartenente al set della DBMIRROR classica (comprendente anche le tabelle delle videate personalizzate).

Diversamente, con <tip>=1 si stabilisce che <rif$> è il nome di un file di personalizzazione in cui sono codificate per ciascuna tabella le colonne che devono essere trattate nell'esportazione (se il file contiene il riferimento a più tabelle l'esportazione avverrà per ognuna di esse). Per i parametri <dir$> <file$> vedere ASCMIRROR.

Il nome del file di personalizzazione richiede quello della tabella e della colonna separati da uno o più blank o tabulatore ASCII; le colonne nel file possono avere un ordine diverso da quello definito nella struttura della tabella nel database. L'ordine può essere casuale (tabelle intervallate da linee vuote) e sono inseribili dei commenti (marcatura con apice "'" al inizio di una nuova riga).

Col parametro numerico [vis] facoltativo, si stabilisce il modo di visualizzazione dello stato di avanzamento progressivo dei record trattati (letti e scritti): se = -2 non visualizza, se = -1 visualizza al termine dell'elaborazione il totale dei record trattati, se = 1 o = 0 (o assente) visualizza la progressione dei record (step=1), se = N visualizza con step=N e al termine visualizza il totale di record trattati.

# ASCMIRROR_EXT2

### ASCMIRROR_EXT2 <tip>,<rif$>,<dir$>,<file$>,<slot>,<att>,<var>,<azz>,[impo$]

La [impo$] di ASCMIRROR_EXT2 consente di inserire una stringa per impostare le informazioni nel file ASCII da generare, tramite i seguenti prefissi:

SV = step di visualizzazione (equivalente al parametro [vis])

CI = carattere separatore item (es. ;)

CT = carattere delimitatore delle informazioni di tipo testo (es. ")

Vedere ASCMIRROR_EXT

# ATTIVAFIN

### ATTIVAFIN <nfin>

In Sprix/Collage attiva la finestra <nfin> precedentemente aperta (numero da0 - base del gestionale - a 9; in collage è anche negativo per finestre del gestionale) e posiziona il cursore in riga 1 e colonna 1. Disattiva finestre precedenti (solo una <nfin> alla volta è attiva per le istruzioni a video).

# AUTOFILE

### AUTOFILE()

In Sprix/Collage ritorna il primo numero di file disponibile; se nessun numero è disponibile, la funzione ritorna 0.

# AUTOFIN

### AUTOFIN()

In Sprix/Collage ritorna il primo numero di finestra disponibile (0=nessuno)

# AZZARRAY

### AZZARRAY [num/str$]

Azzera il dimensionamento dell'array effettuato con DIM

Esempi di sintassi di 3 dimensionamenti array numerico 'N' e stringa 'A$'

| | una dimensione | due dimensioni | tre dimensioni |
|---|---|---|---|
| AUTOMATICI: | DIM N() | DIM N(,10) | DIM N(,10,20) |
| NON AUTOMATICI: | DIM A$(5) | DIM A$(5,10) | DIM A$(5,10,20) |

Si accede a ciascun elemento alla volta tramite l'indice fra parentesi tonde indicato in modo diretto o ciclico, anche tramite variabile.

```
Esempi:
DIM A$(3): A$(1)="pass", assegna al primo di 3 elementi la stringa "pass";
DIM N():
AZZARRAY N
FOR I=1 TO 5:
        N(I)=I+1:
NEXT
assegna 5 valori (1+1, 2+1, 3+1, 4+1,5+1) che assume l'indice ai 5 elementi che automaticamente
ne conseguono ed a cui successivamente poter accedere e gestire con indice diretto (N(3)=4),
indice variabile (X=2: N(X)=3) o ciclico (con indice X da 1 a 5 per ottenere N(1)=2, N(2)=3, N(3)=5,
N(4)=5, N(5)=6 );
```

# AZZCACHE

Istruzione Sprix/Collage per la gestione della CACHE relativa al Dizionario.

Se non viene eseguita, non avviene l'aggiornamento in tempo reale delle variabili Dizionario. Le variabili di struttura sono invece gestite attraverso  le apposite AZZVARSYS relative a ciascuna singola struttura.

# AZZCRONO

### AZZCRONO <nc>

In Sprix/Collage legge il cronometro numero <nc> e ritorna il tempo trascorso in forma "secondi.millesimi", dall'ultimo azzeramento con AZZCRONO <nc>.

Sono consentiti fino a 3 cronometri contemporaneamente.

```
Esempio
AZZCRONO 1
CICLO: PRINT CRONO(1): IF CRONO(1)<10 THEN GOTO CICLO
```

# AZZTAST

In Sprix/Collage cancella i caratteri in sospeso nel buffer di tastiera, compreso quelli inviati con PUTTAST.

# AZZVARSYS

### AZZVARSYS ‘n’

In Sprix azzera variabili di struttura archivi “n” identifica l’archivio secondo la tabella indicata

| | | |
|---|---|---|
| 1 | _PC? | Anagrafica Piano dei Conti. |
| 2 | _PN? | Movimento di prima nota. |
| 3 | _AR? | Anagrafica Articoli. |
| 4 | _MM? | Documento di magazzino |
| 5 | _IL? | Impegni di Lavorazione |
| 6 | _OB? _OL? | Bolla di Lavorazione |
| 7 | _PART? | Particolarità Conti e Articoli. |
| 8 | _TP? | Distinta Base Primaria. |
| 12 | _DA? | Anagrafica Indirizzi di Spedizione. |
| 13 | _DC? | Anagrafica Contatti |
| 17 | _LOT? | Anagrafica Lotto. |
| 18 | _TPLOT? | Anagrafica Tipo Lotto. |
| 19 | _CDG? | Movimenti di Analitica del controllo di gestione |
| 20 | _DV? | Documenti Docuvision |
| 21 | _AL? | Codici Alias |
| 22 | _AB? | Codici Abbinati. |
| 23 | _CO? | Anagrafica Commesse; |
| 24 | _AE? | Anagrafica Aree. |
| 25 | _AD? | Associazioni di Documenti Gestionale-Docuvision. |
| 33 | _CFR? | Referenti della Rubrica Clienti / Fornitori. |
| 34 | _AGP? | Agenda di Lavoro. |
| 35 | MYDB | _MYDBELE, _MYDB$(), _MYDBANN$, _MYDBETIC$(), _MYDBID, _MYDBNR e _MYDBARC$ |

# BEEP

In Sprix/Collage emette un suono per richiamare l'attenzione. Richiede settaggio di apposita frequenza nel file ".ini" di configurazione.

# BLOCCASEM

### BLOCCASEM <nomesem$>

In Sprix/Collage blocca il semaforo con nome <nomesem$> (fino a 8 caratteri numerici e/o lettere maiuscole). Il blocco è globale per tutti i terminali collegati al sistema (e quindi non solo relativamente alla medesima azienda aperta) se <nomesem$> inizia con '*'. Ogni terminale può bloccare contemporaneamente max 10 semafori (di cui massimo 3 globali). BLOCCASEM valorizza _ERRSEM che riporta il codice dell'esito dell'operazione BLOCCASEM e indica lo stato del semaforo:

| 0 | è stato bloccato (anche con sistema composto da un solo terminale); |
|---|---|
| 1 | non è stato bloccato (è già bloccato da un altro terminale); |
| 2 | non è stato bloccato (<nomesem$> errato); |
| 3 | non è stato bloccato (sono in uso troppi semafori); |
| 4 | non è stato bloccato (sono in uso troppi semafori globali); |

```
Sprix1
BLOCCA:BLOCCASEM "*SEM1"
IF _ERRSEM THEN VIMSG 4, "Sem. bloccato da Sprix2/Collage":GOTO BLOCCA
PRINT "posso scrivere...";:INPUT "A",A$,1:SBLOCCASEM "*SEM1":END
```

# CALL

### CALL <nomefile$>,<nomeetic$>

In Collage richiama e attiva l'esecuzione della routine stabilita nell'etichetta <nomeetic$> del nome file Collage senza estensione <nomefile$> e che sia già caricato in memoria (con AUTO con STARTCOLLAGE), pena l'interruzione della routine con messaggio "nomefile$.es non caricato in memoria". CALL sospende l'esecuzione del Collage caricato in memoria finchè la routine richiamata non termina con END per poi riprendere; è simile alla GOSUB, ma viene terminata con END anzichè RETURN e <nomeetic$> può appartenere a un altro Collage.

# CAMBIOVAL

### CAMBIOVAL(<valp>,<vala>,<impo>,<dtcambio>,<numdec>)

In Sprix/Collage converte l'importo <impo> (accetta anche 0) espresso in valuta <valp> (da 1 a 30, altrimenti ritorna 1) nella valuta <vala> richiesta (da 1 a 30, altrimenti ritorna 1), alla data del cambio <dtcambio> definita (se non è valida viene assunta l'odierna) e con num. di decimali <numdec> richiesto (da 0 a 6; se <0 è assunto 0, se >6 è assunto 6)

# CARCOLLAGE

### CARCOLLAGE(<nomefile$>)

In Collage ritorna il numero di ordine nell'esecuzione del Collage caricato in memoria con nome file <nomefile$> senza estensione; torna 0 se il file non è ancora caricato.

# CHIUDIFIN

### CHIUDIFIN <nfin>

In Sprix/Collage chiude il numero di finestra <nfin> da 1 a 9 (esclusa la 0)precedentemente aperta e attiva quella aperta immediatamente prima con posizione del cursore in riga 1, colonna 1. Chiude le finestre aperte successivamente e indipendentemente dalla sequenza, anche a caso (nfin èindicativo e non indica la sequenza. Apertura 6,3,5,2 Chiusura 5: chiude 5, 2 e attiva 3

NOTARE. La sequenza di APRIFIN/CHIUDIFIN deve rispettare la gerarchia fra le finestre nelle seguenti regole: la chiusura di una finestra comporta la chiusura delle figlie; nell'intervallo fra l'apertura e la chiusura della finestra da parte del gestionale (es. l'estensione della descrizione sulla riga articolo) non può essere chiusa quella Collage (quella collage può vivere solo al interno o al esterno di questo intervallo.

```
Esempio:
momento T1 Aprifin Collage
momento T2 Aprifin Gestionale
momento T3 Chiudifin Gestionale
momento T4 Chiudifin Collage
Se ciò non è rispettato la routine collage si interrompe e rilascia errore per impossibilità di
chiusura finestra.
```

# CHIUDISTAMP

### CHIUDISTAMP

Chiude la stampante precendetemente aperta.

# CHR

### CHR$(<num>)

In Sprix/Collage ritorna una stringa equivalente al valore ASCII di <num> (da 0 a 255).

Esempio:

X=87: il risultato di CHR$(X) è "W"

# CIFLET

### CIFLET$(<num>)

In Sprix/Collage converte il numero <num> in lettere senza trattare il segno e decimali.

Esempio:

X=-1234.88: CIFLET$(X) risultato: MILLEDUECENTOTRENTAQUATTRO

# CIFLETDEC

### CIFLETDEC$(<num>,<numdec>)

In Sprix/Collage converte il numero <num> (con decimali separati da '.') in lettere con decimali separati da '/' e stabiliti in <numdec> (da 0 a 6).

Esempio:

Y=6543.21: CIFLETDEC$(Y,3) risultato: SEIMILACINQUECENTOQUARANTATRE/210

Valori Euro si convertono nella parte intera e i due decimali (anche se =00)

# CLOSE

### CLOSE <numfile>

Terminate le operazioni della elaborazione del file successive all'apertura, CLOSE <numfile> in Sprix/Collage chiude il file numero <numfile> per renderlo disponibile. Il termine del programma chiude il file.

# CLOSECOM

Chiude la porta precedentemente aperta per la lettura/scrittura. L'esecuzione su Client può richiedere reindirizzamento "<com$".

# CLS

In Sprix/Collage cancella tutto il contenuto della finestra attiva (se è la base -0- le righe riservate 1,2,3,24,25 dello schermo fisico sono escluse).

Le coperture a video avvengono con finestre con gestione del posizionamento del cursore all'uscita dei campi. La pulizia di campi sporchi avviene con la chiusura e riapertura della finestra.

# COLOCBL

### COLOCBL <NumBL>,<NumPF>,<NumRecRigaOC>

Consente in Sprix il collegamento di una riga OC ad una BL; se il numero di Prodotto Finito <NumPF> non viene indicato (=0), la riga OC relativa al record <NumRecRigaOC> viene aggiunta al numero BL <NumBL>; se <NumPF> è <> da 0, <NumRecRigaOC> viene agganciata al PF con il <NumPF> specificato relativo al nuemro BL <NumBL>; controlli effettuati: che <NumBL> esista; che <NumPF> esista in <NumBL>; che <NumRecRigaOC>sia un record valido, la riga sia sospesa e l'articolo di produzione; se indicato <NumPF>, che i 2 codici siano uguali.

La COLOCBL è controllabile con la _ERRCOLOCBL e _ERRCOLOCBL$ che in Sprix riportano il codice e la descrizione dell'esito dell'operazione:

| | |
|---|---|
| 0 | esito positivo dell'operazione; nessun messaggio, il campo è vuoto; |
| 1 | generico con relativa descrizione:<br>• riga OC non valida;<br>• riga già collegata a XX xx/xxxxxx;<br>• l'articolo ???? è senza DBP;<br>• codici incompatibili: '????' e '????';<br>• incompatibilità tipo articolo/quantità;<br>• documento in uso da altro terminale riga non modificabile;<br>• riga già variata da altro terminale;<br>• riga con distinta base automatica riga non modificabile;<br>• riga bloccata riga non modificabile;<br>• riga ordine cliente annullata riga non modificabile; |

| | |
|---|---|
| | • errore generico riga non modificabile; |
| 2 | numero bolla errato; |
| 3 | memoria insufficiente; |
| 4 | bolla padre non trovata; |
| 5 | bolla non trovata; |
| 6 | errore generico; |

# COLOR

### COLOR <colore>

In Sprix/Collage imposta il colore di cursore e caratteri da visualizzare a video in base al numero <colore>: 0=normale; 1=reverse; 2=alta intensità; 3=riga evidenziata (tipo a tendina); 4=campo di immissione. Il colore effettivo è determinato da sprix in funzione al colore di sfondo della finestra video.

Esempio:

```
A=2: COLOR A: PRINT "alta intensità"
```

# COPYFILE

### COPYFILE <nomefile1$>,<nomefile2$>

In Sprix/Collage copia il nome file <nomefile1$> nel nome file <nomefile2$> in cui sostituire i nuovi dati al posto dei precedenti. I file devono esistere ed essere chiusi, possono essere preceduti dalle rispettive directory. Se<nomefile1$> non esiste la copia non è effettuata, se <nomefile2$> non esiste viene creato.

# CRONO

### CRONO(<nc>)

In Sprix/Collage legge il cronometro numero <nc> e ritorna il tempo trascorso in forma "secondi.millesimi", dall'ultimo azzeramento con AZZCRONO <nc>.

Sono consentiti fino a 3 cronometri contemporaneamente.

Esempio

```
AZZCRONO 1
CICLO: PRINT CRONO(1): IF CRONO(1)<10 THEN GOTO CICLO
```

# CTRCODFIS

### CTRCODFIS(<codfis$>)

In Sprix/Collage controlla il Codice Fiscale del conto <codfis$>; ritorna 0 se errato, 1 se corretto. Es. CTRCODFIS(pacfi("201.00003"))

# CTRPIVA

### CTRPIVA(<partiva$>)

In Sprix/Collage controlla la Partita Iva del conto <partiva$>; ritorna 0 se errata, 1 se corretta. La P.Iva è numerica (al massimo preceduta dai caratteri "IT", altrimenti èconsiderata straniera - valore = 1 -).

# CVA

### CVA$(<str$>,<numbyte>)

In Sprix/Collage toglie dalla stringa <str$> lunga <numbyte> tutti i caratteri ASCII=0 partendo dalla destra fermandosi al primo carattere non zero.

Esempio:

```
B$="Sprix  ": C$=CVA$(B$,7) risultato C$="Sprix"
```

# CVMAIU

### CVMAIU$(<str$>)

In Sprix/Collage converte la stringa <str$> in maiuscolo.

Esempio

```
A$="Sprix"
```

B$=CVMAIU$(A$) o B$=CVMAIU$("Sprix") risultato B$="SPRIX"

# CVMINU

### CVMINU$(<str$>)

In Sprix/Collage converte la stringa <str$> in minuscolo.

Esempio C$="SpriX"
D$=CVMINU$(C$) o D$=CVMINU$("SpriX") risultato D$="sprix"

# CVN

### CVN(<str$>,<numbyte>,<numdec>)

In Sprix/Collage converte in numero con <numdec> decimali la stringa <str$> lunga <numbyte>, impaccata precedentemente con MKN$.

Esempio
C=CVN(C$,2,0)  interpreta i primi 2 byte di C$ per ottenere il numero C

# DBMIRROR

### DBMIRROR <tabella$>

Effettua la lettura di tutti i dati contenute nei record del nome di <tabella$> del dizionario (disponibile in FTP directory del disco "**ftp.passepartout.sm/Aggiornamenti/Conces/**"), esportando in sovrascrittura/precancellazione del supporto queste informazioni su file RDBMS(DBMIRROR).

Richiede la configurazione ODBC DRIVER MANAGER da specificare nella funzione Configurazione Moduli[F4] dei dati Anagrafici aziendali; I campi data non valorizzati nel gestionale sono convertiti al 1/1/1900

Viene visualizzato lo stato di avanzamento progressivo dei record trattati e si può interrompere con tasto [Esc] (previa impostazione di _INTER) con riempimento del file/database fino a quanto eseguito (l'interruzione dell'elaborazione della tabella corrente non interrompe gli eventuali mirroring successivi). Non è eseguibile in ciclotabellare, di cui valgono alcune regole, come LEGGIDESC per l'articolo descrittivo nel documento, l'elaborazione annuale per anno di ingresso in azienda;

> *Attenzione: il mirroring (anche quello in tempo reale) dei documenti di anni diversi con chiave identica (ma che non comprenda l'anno, es. sigla+serie+numero+conto a numeratore unico e non annuale) si sovrascrivono.*

Il mirroring di interi archivi impegna il sistema, se è già appesantito da altre condizioni (dbmirror in tempo reale, reti sovraccariche, computer non performanti..) è preferibile l'export delle tabelle necessarie ai soli dati di interesse da esportare con procedure batch di fine lavoro oppure interrogazioni di database replicati in chiaro o infine mirroring ottimizzati.

Il mirroring non va eseguito nel ciclotabellare.

# DBMIRROR_EXT

### DBMIRROR_EXT <tip>,<rif$>,<slot>,<att>,<var>,<azz>,[vis]

| | |
|---|---|
| <tip> | = 0 si stabilisce che il riferimento <rif$> è il nome di una tabella appartenente al set della DBMIRROR classica (comprendente anche le tabelle delle videate personalizzate).<br>=1 si stabilisce che <rif$> è il nome di un file di personalizzazione in cui sono codificate per ciascuna tabella le colonne che devono essere trattate nell'esportazione |
| <rif$> | nome di un file di personalizzazione in cui sono codificate per ciascuna tabella le colonne che devono essere trattate nell'esportazione |
| <slot> | un numero da 1 a 8, slot di esportazione presente nella  tabella Stato anagrafiche del Gestionale |
| <att> | = 0 tratta tutti i record,<br>= 1 solo se sono attivi |
| <var> | = 0 non gestisce ottimizzazioni di export<br>= 1 esporta i soli record variati dall'ultimo aggiornamento<br>= 2 elimina i record dal database non gestendo lo stato del variato dello <slot> |
| <azz> | = 0 non azzera lo stato variato<br>= 1 azzera lo stato variato |
| [vis] | = -2 non visualizza<br>= -1 visualizza al termine dell'elaborazione il totale dei record trattati<br>= 1 o = 0 (o assente) visualizza la progressione dei record (step=1) |

In Sprix ottimizzano la DBMIRROR sulle tabelle Anagrafiche Piano dei Conti e Articoli, riducendo il volume dei dati esportato dalle tabelle:

PC_ PP_ PA_ e AR_ AH_ AF_ AX_ AY_ AL_ AB_ AW_ AQ_ AO_ AE_.

La riduzione avviene con i flag <slot>,<att>,<var>,<azz>. Nel menu Aziende -Parametri di base - tabella Stato anagrafiche del Gestionale, si possono stabilire fino a 8 tipi di slot di esportazione (da utilizzare in altrettante applicazioni Sprix (esempio E-COMMERCE) per le quali definire, con la pressione del tasto [SHF11] e la selezione di 'Stato anagraficà sul conto o sul articolo nelle rispettive anagrafiche, lo stato ATTIVO oppure NON ATTIVO dello<slot> sulla variazione del dato: il singolo record è quindi esportato o meno in base a quanto stabilito con lo <slot> sul flag di variazione del singolo record (che viene alzato dal Gestionale alla modifica del record). Nella operazione PUT? la scrittura default è N per 'ATTIVÒ e S per il 'VARIATÒ;gli elementi 9-12 sono invece di sola lettura.

Lo <slot> è quindi un numero da 1 a 8; se = 0 non gestisce questa ottimizzazione attraverso il flag numerico <var> per l'export dei soli record variati dall'ultimo aggiornamento (se <var>=1) o di tutti i record con cancellazione (se <var>=0) o drop (se <var>=2) della tabella nel database (rendendo quindi irrilevante il flag 'Variatò dello <slot>). La Configurazione Moduli ([F4] in Anagrafici Azienda) contiene un parametro 'RIMOZIONE DATI CANCELLATI' che DBMIRROR_EXT consente di tenere traccia in un file log dbmi.sig (sig = sigla azienda) dei record rimossi nei movimenti di magazzino per il successivo allineamento quando si opera per differenza. Gli <slot> della DBMIRROR e ASCMIRROR sono in comune: una esportazione su database con DBMIRROR determina la non esportazione su file dell'ASCMIRROR sul medesimo slot.

Il <var> dipende comunque dal flag numerico <att> (se <att>=0 tratta tutti i record, se <att>=1 solo se sono attivi). Terminata l'esportazione, per evitare successivi ri-trattamenti del record, il <var> di tutti gli slot (relativi al intero archivio anagrafico e non della singola tabella) viene azzerato se <azz>=1 (se <azz>=0 non azzera <var>), poichè se il flag numerico <tip> è = 0 si stabilisce che il riferimento <rif$> è il nome di una tabella appartenente al set della DBMIRROR classica (comprendente anche le tabelle delle videate personalizzate).

Diversamente, con <tip>=1 si stabilisce che <rif$> è il nome di un file di personalizzazione in cui sono codificate per ciascuna tabella le colonne che devono essere trattate nell'esportazione (se il file contiene il riferimento a più tabelle l'esportazione avverrà per ognuna di esse) e in cui è la medesima DBMIRROR_EXT a preoccuparsi di alzare il flag quando si elabora l'ultima tabella dell'archivio.

Il nome del file di personalizzazione richiede quello della tabella e della colonna separati da uno o più blank o tabulatore ASCII; le colonne nel file possono avere un ordine diverso da quello definito nella struttura della tabella nel database. L'ordine può essere casuale (tabelle intervallate da linee vuote) e sono inseribili dei commenti (marcatura con apice "'" al inizio di una nuova riga).

Se col file di personalizzazione è possibile indicare quali colonne trattare (selezione in verticale di un'ipotetica matrice), con gli slot è possibile filtrare i record da trattare con DBMIRROR_EXT (selezione in orizzontale)

Col parametro numerico [vis] facoltativo, si stabilisce il modo di visualizzazione dello stato di avanzamento progressivo dei record trattati (letti e scritti): se = -2 non visualizza, se = -1 visualizza al termine dell'elaborazione il totale dei record trattati, se = 1 o = 0 (o assente) visualizza la progressione dei record (step=1), se = N visualizza con step=N e al termine visualizza il totale di record trattati.

# DELABBIN

### DELABBIN <codart$>,<codab$>,<nprog>,<cat$>

In Sprix cancella i dati dei Codici Abbinati relativi alle variabili di struttura archivi _AB? in base al codice articolo <codart$> a cui sono abbinati (se "" = tutti gli articoli), al codice abbinato <codab$> (se "" = tutti gli abbinati), al numero progressivo per codici abbinati doppi <nprog> (se -1 = tutti, altrimenti specificare il numero di chiave che si intende prelevare: 0,1,2,..) e alla categoria di abbinamento <cat$> (se "" = tutte le categorie di abbinati). Non è consentita la cancellazione di tutti gli alias per tutti gli articoli.

La lettura rilascia _NAB cioè il numero di elementi dell'array dinamico della struttura _AB?

# DELAGP

### DELAGP [nrec],[dal$]

In Sprix cancella i dati delle righe della AGENDA DI LAVORO, relativi alle variabili di struttura archivi _AGP?, in base ai seguenti parametri identificativi obbligatori:
**nrec** = Num. record (consente la ricerca diretta o la simulazione del ciclo)
**dal$** = Da data lavoro "AAAMMGG"

# DELALIAS

### DELALIAS <codart$>,<codalis$>,<nprog>

In Sprix cancella (DEL) i dati dei Codici Alias relativi alle variabili di struttura archivi _AL? sulla base al codice articolo <codart$> a cui sono associati (se "" =tutti gli articoli), al codice alias <codalias$> (se "" = tutti gli alias) e al numero progressivo per i codici alias doppi <nprog> (se -1 = tutti, altrimenti specificare il numero di chiave che si intende prelevare: 0,1,2,..). Non è consentita la cancellazione di tutti gli alias per tutti gli articoli.

La lettura rilascia la _NAL per conoscere il numero di elementi dell'array dinamico della struttura _AL?

# DELAR

### DELAR <codart$>

In Sprix cancella i dati di Anagrafica Articoli relativi alle variabili di struttura archivi _AR? in base al codice articolo principale <codart$>

# DELASSOCDV

### DELASSOCDV <tipo>,<keypass>,<iddoc>

In Sprix cancella i dati delle associazioni fra il Documento Docuvision e l'Entità del Gestionale, relativi alle variabili di struttura archivi _AD?, in base all'ID del documento Docuvision <idDoc> se <idDoc>=0 le elimina tutte e il <tipo> di Entità del Gestionale, determinata univocamente dalla <keypass> di ricerca sulle tabelle d'associazione fra Gestionale-Docuvision. Le Entità del Gestionale appartenenti ad archivi annuali devono essere relative all'anno aperto.

Il parametro facoltativo [alleg] consente di variare la tipologia di allegato e il progressivo immagine attraverso la PUTASSOCDV. Il parametro ha la seguente sintassi "TPALL=x;PRGALL=y" dove "x" può assumere i seguenti valori null=Generico o I=Immagine o S=Scheda) o D=Descrizione e dove "y" può assumere valore numerico relativo al progressivo. Essendo parametro facoltativo,l'omissione di [alleg] equivale al default Generico "TPALL=;". Esempio di cambio di allegati Generici (associati ad "ART") in allegati immagine:

```
GETASSOCDV 1200,"ART",0
FOR X=1 TO _ADNUMASSOC
PUTASSOCDV 1200,"ART1",_ADDVDID(X),"TPALL=I;PRGALL="+STR$(X,0,0)
NEXT
```

GETASSOCDV valorizza _ADNUMASSOC che ritorna il num. globale d'associazioni.

La GET-PUT-DEL elaborano i dati _AD? in base ai seguenti parametri (eccetto per le classi <tipo>=1 Stampa generica, =800 Dichiarativi/Versamenti, =900 Stampe Fiscali; =1000 Agenda di lavoro ed eccetto i casi in cui il livello di azienda o il codice contratto non consentono l'accesso ad alcune tabelle,come es le operazioni di classe 'Parcella/Notulà senza gestione parcellazione, classi di Magazzino con azienda Contabile,...):

```
Piano dei Conti: <tipo>=400
  <keypass>="<codcon$>" (codice conto del piano dei conti)

Primanota: <tipo>=600
  <keypass> = "<dreg$>;<prinr>;<cau$>;<protr$>;<prots>;<protn>;<docn>;<ddoc$>"
                    <dreg$> (data di registrazione);
                    <prinr> (n. progressivo registrazione primanota);
                    <cau$>  (causale del documento);
                    <protr$>(registro di protocollazione)
                    <prots> (serie del protocollo)
                    <protn> (numero di protocollazione)
                    <docn>  (numero del documento)
                    <ddoc>  (data documento)

Anagrafica Articoli: <tipo>=1200
  <keypass> = "<codart$>" (codice articolo)

Movimenti di Magazzino: <tipo>=1400
  <keypass> = "<sig$>;<ser>;<num>;<clifor$>;<ddoc$>"
                    <sig$>    (sigla documento)
                    <ser>     (serie del documento)
                    <num>     (numero del documento)
                    <clifor$> (codice conto intestatario)
                    <ddoc$>   (data documento)

Ordine/Preventivi/Matrici: <tipo>=1500
  <keypass>="<sig$>;<ser>;<num>;<ddoc$>"
                    <sig$>  (sigla documento)
                    <ser>   (serie del documento)
                    <num>   (numero del documento)
                    <ddoc$> (data documento)

Bolla di Lavoro: <tipo>=1600
  <keypass> = "<numeroBL>;<numeroSottoBL>"
                    <numeroBL>      (numero di bolla di lavoro)
                    <numeroSottoBL> (numero di sottobolla di lavoro)
                    Inserire "-1" per avere tutte le sottobolle, inserire "0" (o non specificare) per intendere la BL principale. In
                    operazione DEL è richiesta anche la data della bolla.

Notule/Parcelle: <tipo>=1100
  <keypass>="<cau$>;<ser>;<num>;<clifor$>;<ddoc$>"
                    <cau$>    (causale del documento)
                    <ser>     (serie del documento)
                    <num>     (numero del documento)
                    <clifor$> (codice conto)
                    <ddoc>    (data documento)
```

# DELBL

### DELBL <numeroBL>,<numeroSottoBL>

In Sprix cancella la Bolla di Lavorazione in base al numero della bolla <numeroBL> e sottobolla <numeroSottoBL> di lavorazione (per le sottoBL dei lavoranti aggiungere 10000, es. la sottobl 0002->10002).

Il Collage legge le _OB? e _OL? con la GETATTBL (senza parametri): azzera le variabili preesistenti e carica dati presenti a video fra evento START e END e fra PUT e DEL.

La cancellazione di una BL senza indicare le sottoBL apre una finestra per confermare la cancellazione delle sottoBL collegate alla BL.

## DELCDG

### DELCDG <TipoDoc$>,<KeyDoc$>,<Anno>,<NumRigaDoc>,<NumProgAna>

Cancella i dati dei movimenti di Contabilità Analitica relativi alle variabili _CDG?, in base al Tipo movimento <TipoDoc$> (R=Primanota; M=Magazzino; C=Ord.Cliente; F=Ord.Fornitore; P=Preventivi; A=Marici; E=Extra) nell'anno del movimento <Anno> (contabile 1960 -2048 se tipo C/F, fisso 2048 se tipo E/P, fisso 1 se tipo A) al numero della riga del documento <NumRigaDoc> a cui sono associate le registrazioni di analitica (se 0=tutte le righe) e al numero progressivo di riga <NumProgAna> relativo all'analitico (se 0=tutte le righe; in magazzino è sempre 0 dato che per ciascuna riga di movimento è consentita una sola riga di analitica mentre in primanota ne sono consentite N). La Chiave del movimento <KeyDoc$> è formata da campi obbligatori di ricerca in base a <TipoDoc$> (separati da ;)

R - "PNDRE=x;PNNDO=x;" (PNNDO = num.progressivo registrazione primanota);

M - "MMSIG=x;MMNUM=x;MMSAZ=x;MMSER=x;MMCLI=x;NTMM=x;" (M è in relazione all'anno di ingresso; MMCLI è facoltativo per i documenti non intestati;

se NTMM - Num. riferimento testata documento  non è definito è impostato = 0);

C/F/P/A - "MMSIG=x;MMNUM=x;MMSAZ=X;MMSER=x;";

E - "PRGEXTRA=x;DTCOMP=x;AREA=x;COMM=x;CODANL=x;TIPMOV=x;" (il primo parametro PRGEXTRA non è obbligatorio);

La lettura rilascia la _CDGNR per conoscere il numero di elementi dell'array dinamico (numero righe) della struttura _CDG?

## DELCFR

### DELCFR <cod$>,<tipo>

Cancella  i riferimenti dei Referenti presenti nell'anagrafica della rubrica Clienti Fornitori (tasto funzione F4) relativi alle variabili di struttura _CFR? in relazione ai valori del Cliente/Fornitore <cod$> in base al <tipo> (se 0 = tutti i referenti, da 1 a 9 = un referente specifico).

In Collage legge le variabili _CFR? con GETATTCFR (senza parametri); carica i dati presenti a video fra gli eventi ON_STARTCFR e ON_ENDCFR.

## DELCON

### DELCON <codice$>

In Sprix cancella i dati di Anagrafica Contatti relativi alle variabili _DC? in base al codice contatto <codice$>

## DELDB

### DELDB <codart$>,<numfase>

In Sprix cancella i dati della Distinta Base Primaria relativi alle variabili di struttura archivi _TP? in base al numero di fase <numfase> della DBP (1-99) e del codice articolo <codart$> con DBP.

Le operazioni sulla DBP valorizzano _NCOMP (numero dei componenti di una fase: max 5000 per fase).

La cancellazione è per tutte le fasi  se <numfase> = 0 oppure solo dell'ultima se <numfase> <> 0.

## DELDVDOC

### DELDVDOC <numdoc>,<numrev>

In Sprix cancella i dati del Documento Docuvision relativi alle variabili struttura archivi _DV?, in base al numero Documento di Docuvision <numdoc> e numero revisione <numrev> (in lettura se -1=tutte le revisioni, 0=ultima revisione, >0=revisione specificata;in cancellazione se 0=cancella l'intero documento).

# DELIL

### DELIL ndoc, nsot, nropf, fase, mag, cod

| | |
|---|---|
| ndoc | Numero della bolla di lavorazione, corrisponde alla variabile _ILNDO |
| nsot | Numero della sotto bolla di lavorazione, corrisponde alla variabile _ILNSD |
| nropf | Numero riferimento prodotto finito, corrisponde alla variabile _ILNRB |
| fase | Numero di fase del componente, corrisponde alla variabile _ILFAS |
| mag | Magazzino materie prime, corrisponde alla variabile _ILMAG |
| cod | Codice del componente, corrisponde alla variabile _ILAMP$ |

In Sprix cancella i dati degli Impegni di Lavorazione relativi alle variabili di struttura archivi _IL?.

Richiede obbligatoriamente il numero della Bolla di Lavorazione <ndoc>, il numero della sottobolla <nsot> e il numero di rifererimento del prodotto finito <nropf>;

la specificazione del codice della materia prima <cod> (se "" = tutte le MP) implica l'indicazione del numero di magazzino <mag> (se 0 = tutti i magazzini) che, se specificato a sua volta implica il numero fase <fase> (se 0 = tutte le fasi) . Per effetutare la cancellazione dei residui di scarico(-1=RES_SL) o residui di deposito (-2=RES_DL) il num. di rifer. prod. finito <nropf> non potendo essere un numerio negativo, indicare per RES_SL il valore 65534 al posto di -1 e per RES_DL 65533 al posto di -2.

*N.B.: Non è possibile cancellare tutto l'archivio impegni, o tutti gli impegni di una bolla di lavorazione con una sola istruzione DELIL*

```
Esempio di cancellazione:
Leggo tutti gli impegni della bolla di lavorazione.
GETIL 149,0,0,0,0,""
Ciclo da 1 a quanti impegni ho letto
FOR I = 1 TO _NIL
Cancello tuttli gli mpegni con la chiave che ho letto precedentemente.
        DELIL 149,0,_ILNRB(I),_ILFAS(I),_ILMAG(I),_ILAMP$(I)
        IF _ERRIL <> 0 THEN VIMSG 5,_ERRIL$+"| DELIL"
NEXT
```

# DELKEY

### DELKEY <numfile>,<key$>

In Sprix/Collage nel file a indici aperto con numero <numfile> cancella in base alla chiave stringa principale <key$>; se la chiave non esiste la DEL non esegue nessuna operazione; tutte le chiavi sono obbligatorie e i loro valori non possono essere tutti ASCII 0 o ASCII 255. La Cancellazione di chiavi diverse possono richiedere tempi differenti in base a dimensione/struttura del file, posizione logica della chiave. Valorizzano _NRDATI e _ERRIND.

# DELLOT

### DELLOT <idlot>

In cancella i dati di Anagrafica Lotto relativi alle variabili struttura archivi _LOT? in base al numero ID lotto <idlot>. La cancellazione avviene se non è in uso da utente e/o non è stato caricato da movimento magazzino (in pratica è cancellabile se in anagrafica lotti è attivo il tasto [ShF3] per attivare la cancellazione dal modulo di servizio).

# DELMM

### DELMM <doc$>

In Sprix cancella il Documento di Magazzino in base al documento <doc$> in forma SIGLA[SERIE/]NUMERO (o SIGLA/NUMERO + SIGLA/NUMERO+ecc se più di uno); la sintassi identifica documenti dell'anno solare dell'azienda aperta (per l'infrannuale, il parametro di magazzino "Numerazione solare mov.mag(S/N): N" consente di inserire l'anno del documento "nn" prima del protocollo SIGLASERIE/nn-NUMERO, es. GETMM "FT5/08-3432" - per la infrannuale aperta nel 2009 - o GETMM "FT5/09-3432" - per quella aperta nel 2008 -)

Il Collage legge le _MM? con GETATTMM (senza parametri): azzera le variabili preesistenti e carica dati presenti a video fra evento START e END e fra PUTe DEL (lo START si attiva dopo l'uscita dal primo campo che divide l'emissione dalla revisione, mentre in emissione differita si attiva con trasformazione del documento e non con la ristampa).

# DELMM_EXT

### DELMM_EXT <doc$>[,conto$[,cancpn]]

L'identificazione del documento presente in archivio più di una volta ma di clienti/fornitori diversi apre una finestra di selezione del documento e la finestra non viene aperta e viene cancellato il primo documento trovato: la DELMM_EXT consente di indicare il conto e in più di stabilire con [,cancpn]= 1 l'eventuale cancellazione di primanota collegata al documento di magazzino (se la primanota non è cancellabile il documento non è cancellato).

# DELPART

### DELPART <tppart$>,<codcon$>,<catcon>,<codart$>,<catart>,<dtvalda$>

In Sprix cancella (DEL) i dati delle Particolarità Conti e Articoli relativi alle variabili struttura archivi _PART?,in base al tipo particolarità <tppart$>(P,S,V,A), al codice conto <codcon$>(o in alternativa al suo numero di categoria <catcon> listino, sconto, provvigione), al codice articolo <codart$> (o in alternativa al suo numero di categoria <catart> formazione prezzo, sconto, provvigione) e con data d'inizio validità <dtvalda$> della particolarità. La combinazione di quattro parametri individua la specifica particolarità e di questi, il secondo e il terzo sono combinabili in quattro modi:(<codcon$>,<codart$>); (<codcon$>,<catart>); (<catcon>,<codart$>); (<catcon>,<catart>).

Collage legge anche con GETATTPART (senza parametri): azzera le variabili preesistenti e carica i dati presenti a video fra evento START e END e fra PUT e DEL.

# DELPC

### DELPC <codcon$>

In Sprix scrive (PUT) legge (GET) cancella (DEL) i dati del Piano dei Conti relativi alle variabili di struttura _PC? in base al codice conto <codcon$>.

La PUTPC del conto nuovo richiede i dati obbligatori della versione del gestionale in uso; la PUTPC di un conto già esistente lo aggiorna (valorizzare sempre tutte le variabili di interesse, altrimenti alcune potrebbero azzerarsi). Prestare anche attenzione alle informazioni relative al caricamento dei conti tramite la trasmissione CSV.

ATTENZIONE: in ambiente a file separati (tecnologia "BP"), per le aziende di gruppo, la PUTPC non è fruibile sui conti diversi da quelli di tipo C/F/B, perchè gli unici conti che questa processa sono i clienti, fornitori e banche (cioè quelli propri dell'azienda in questione). Tramite la variabile _PCSOLOPICO$ = "S/N" (con default = "N") si possono aggiornare tutte le variabili tipiche del conto aziendale anche per i conti di raggruppamento.

I dati della rubrica (RUDT) rimangono invariati. Per i SOLI conti presenti nell'azienda le variabili che possono essere modificate quando la variabile _PCSOLOPICO$ è impostata = "S", sono contrassegnate con il simbolo (*a) nella tabella di variabili _PC? elencate alla fine del paragrafo.

# DELPN

### DELPN <dreg$>,<prinr>,<cau$>,<protr$>,<prots>,<protn>,<docn>,<ddoc$>

In Sprix cancella i dati della registrazione di Primanota relativi alle variabili di struttura archivi _PN?, in base alla data di registrazione <dreg$>, il numero progressivo di registrazione di primanota <prinr> (1-16.777.214), causale del documento <cau$>, registro di protocollo <protr$>, serie di protocollo <prots>, numero della protocollazione <protn>, numero del documento <docn>, data del documento <ddoc$>. La <dreg$>è obbligatoria e con <prinr> forma la chiave primaria di Primanota: l'omissione di <prinr> (se = 0) rende i restanti parametri necessari per identificare e discriminare le registrazioni con dati equivalenti (altrimenti il documento processato è il primo che viene trovato in archivio) e pertanto, devono assumere valore diverso da 0 (se numerico) o "" (se stringa); l'indicazione di alcuni parametri = 0 o = "" può infatti identificare più di un documento, fra i quali il primo incontrato è processato.

Collage legge le variabili _PN? con la GETATTPN (senza parametri): azzera le variabili preesistenti e carica i dati presenti a video fra l'evento START e END e fra la PUT (anche proveniente da registrazioni di magazzino) e la DEL.

Il progressivo <prinr> è contenuto nella _PNPRN di sola lettura e le GET in Sprix/Collage valorizzano _NPN (numero di righe lette nel documento).

# DELTPLOT

### DELTPLOT <tplot$>

In Sprix cancella i dati di Anagrafica Tipo Lotto relativi alle variabili struttura archivi _TPLOT? in base al codice Tipo Lotto <tplot$> (in caso di nuovo tipo lotto indicare "**"), la cancellazione del Tipo Lotto (in stato 'C' creato) richiede il Tipo Lotto <tplot$>.

# DESART

### DESART$(<codart$>)

In Sprix/Collage ritorna la descrizione dell'articolo (comprensiva di quella aggiuntiva) con codice principale <codart$>. Lettere minuscole e caratteri ?nel <codart$> sono ignorati e considerati blank.

# DIFDATA

### DIFDATA(<data1$>,<data2$>)

In Sprix/Collage ritorna la differenza in giorni fra le stringhe <data1$> e <data2$> dal formato "AAAAMMGG" compreso fra 19600101 e 20481231. Il valore ritornato è negativo se <data1$> è < a <data2$>.

Esempio

```
OGGI$="20090620": IERI$=20090619": D=DIFDATA(IERI$,OGGI$)  risultato D = -1
```

# DIM

### DIM (num/str$)([dim1],dim2,dim3)

In Sprix/Collage dichiara le dimensioni [dim1], <dim2> e <dim3> della variabile array <num> (numerico) o <str$> (alfanumerico). Le dimensioni <dim2> e <dim3> sono obbligatorie se dichiarate, mentre la prima [dim1] è facoltativa perchè se non dichiarata si attribuisce una natura automatica all'array che consente risparmio di memoria, grazie all'allocazione fino all'elemento più alto utilizzato (memoria richiesta in automatico per quantità di elementi utilizzati) ma a discapito della velocità.. L'array consente di dichiarare fino a 3 dimensioni (matrice di vettori) da di chiarare con DIM seguito dal nome della variabile array con '()' in cui indicare numeri costanti interi non in forma di variabile, rappresentativi delle dimensioni e del numero di elementi di ciascuna dimensione, fino a un TOTALE MASSIMO di 262080 per l'array stringa e 131040 per il numerico, entrambi ricavati dal PRODOTTO degli elementi di ciascuna dimensione. Indici con valori fuori dalla dimensione dichiarata oppure elementi il cui prodotto eccede il consentito, provocano l'errore 'Indice dell'array fuori dai limiti'. Un numero di elementi non corretti o utilizzati prima del loro dimensionamento (anche quando richiamato in libreria), provocano errori 'dimensioni dell'array troppo grandi', 'array non dimensionato'. L'array non è gestito nella finestra parametri di esecuzione [F6] e non è azzerato nel campo di stampa [F2].

# DISTAST

### DISTAST <dis>

In Sprix/Collage disabilita (<dis>=1) o riabilita (<dis>=0) l'input da tastiera; qualsiasi istruzione di input cancella la disattivazione. Vedere anche _TF e INKEY.

Esempio.
Dato il codice articolo 'ART' e il Collage con la pressione
di F12 sul campo note del menu 'MD':

```
ON_612_MDNOTE: CODART$="A":PUTTAST "#501":DISTAST 1 : DISVID 1
FOR I=1 TO 10: PUTTAST CODART$+"#501#501#501#501": NEXT: END
```

# DISVID

### DISVID <dis>

In Sprix/Collage disabilita (<dis>=1) o riabilita (<dis>=0) il video. Evita all'utente il disturbo da apertura/chiusura di finestre, spostamenti cursore con PUTTAST,... DISVID è cancellata automaticamente da istruzioni input.

# END

Interrompe l'esecuzione del programma Sprix/Collage incondizionatamente e similmente alla pressione di [Esc/Fine] (e con chiusura di ciò che è aperto, file, finestre, stampanti, semafori,...), ma diversamente da quella naturale attraverso l'esecuzione dell'ultima riga di istruzione che non sia del tipo GOTO, GOSUB-RETURN o similari, che conservano le variabili impostate per una successiva esecuzione (ad esempio come nelle elaborazioni multiaziendali se l'azienda elaborata non è l'ultima).

# ESECMENU

### ESECMENU(<menu$>)

In Collage apre automaticamente il menu <menu$> indicato (eccetto 'AF' e nemmeno per lancio di Sprix, che deve avvenire con SHELLSPRIX). In Sprix è consentito lanciare il terminale configurato con un menu sul profilo.

Esempio Collage

```
               Dati cinque Collage di cui quattro sul menu 'MD' e uno su AUTO
   MD           COLLAGE 01          coll_01
  MD           COLLAGE 02          coll_02
  MD           COLLAGE 03          coll_03
  AUTO         COLLAGE 04          coll_04
  MD           COLLAGE 05          coll_05
----------------------------------------------------------------------------
coll_01 (MD)                                 | coll_02 (MD)
----------------------------------------------------------------------------
ON_OUT_MDCAU:                                |ON_IN_MDQTA:
STARTCOLLAGE "coll_05"                       | _GLOBAL(1)=_GLOBAL(1)+1
END                                          | VIMSG 4,"coll_02"
                                             | STOPCOLLAGE "coll_02"
ON_IN_MDRICCOD:                              |END
CALL "coll_05","ETIC_05"                     |-----------------------------
END                                          | coll_03 (MD)
                                             |-----------------------------
ON_OUT_MDTDOC:                               |ON_IN_MDPZZO:
TIPDOC$=CVMAIU$(_SINP$)                      | _GLOBAL(1)=_GLOBAL(1)+1
ABILCOLLAUTO "coll_04",1                     | VIMSG 4,"coll_03"
END                                          | STOPCOLLAGE "coll_03"
                                             |END
ON_IN_MDCODART:                              |-----------------------------
IF TIPDOC$="BC" THEN STARTCOLLAGE "coll_02"  | coll_04 (AUTO)
IF TIPDOC$="FT" THEN STARTCOLLAGE "coll_03"  |-----------------------------
QUANTICOLL=MAXCOLLAGE()                      |ON_IN_MDSCON:
NOMCOLL$=NOMECOLLAGE$(1)                     | VIMSG 4,"Collage AUTO"
NUMCOLL=CARCOLLAGE("coll_01")                | STOPCOLLAGE "coll_04"
_GLOBAL(1)=_GLOBAL(1)+1                      |END
END                                          |-----------------------------
---------------------------------------------| coll_05 (MD)
                                             |-----------------------------
                                             |ETIC_05:
                                             | VIMSG 4,"routine ETIC_05"
                                             |END
                                             |-----------------------------
```

COMMENTO: in emissione documenti di magazzino carica i collage 01, 02, 03 e 05, di cui il numero 01 è in memoria caricato dall'etichetta ON_OUT_MDTDOC:all'uscita della quale è memorizza il Tipo Documento (es. FT) in TIPDOC$ e ABILITA il caricamento del collage AUTOmatico 04 ogni volta che si scatena l'evento sullo sconto in esso contenuto (ON_IN_MDSCON:). Avanzando in uscita dalla causale (ON_OUT_MDCAU:) il coll_01 carica il coll_05 per potere richiamare con CALL la sua etichetta (ETIC_05:) all'ingresso nel campo di ricerca di Cli/For (ON_IN_MDRICCOD:) poichè non sarebbe consentita in quello stessopunto una STARTCOLLAGE "coll_05" dato che le istruzioni Collage sono eseguite all'END dell'evento. Lasciato il campo del Cli/For, all'ingresso del codice articolo (ON_IN_MDCODART:) il coll_01 controlla il valore di TIPDOC$ per stabilire quale fra collage (coll_02 e coll_03) attivare con STARTCOLLAGE in modo che allo scatenarsi dei rispettivi eventi (ingresso in quantità o prezzo) eseguano le routine per poi scaricarsi dalla memoria con la STOPCOLLAGE;lo stesso coll_01 ricava con MAXCOLLAGE() la quantità di collage attivi che sono caricati in memoria, ricava con NOMECOLLAGE$(1) il nome del collage caricato per primo e infine con CARCOLLAGE("coll_01") il numero dell'ordine di posizione del caricamento in Collage e incrementa il primo elemento dell'array numerico _GLOBAL(1) cosi' come lo incrementano coll_02 e coll_03 quando sono caricati.

## ESISFILE

### ESISFILE(<nomefile$>)

In Sprix/Collage controlla l'esistenza del nome file <nomefile$>. Ritorna valore 1 se esiste, altrimenti 0. Gli ambienti Client/Server possono richiedere determinate sintassi di percorsi per non riferire l'operazione di default alla macchina Server (es. "\\...", "<\\...", "<C:\...")

## EXITTAB

Forza l'uscita dal ciclo tabellare (ad esempio quando si verifica una condizione) per proseguire con il codice successivo al ciclo.

## FIELD

### FIELD <numfile>,<lungbuf> AS <varbuf$> [tipoimp], ....

In Sprix/Collage dichiara la struttura del buffer associato al file numero <numfile> aperto con OPEN; la struttura si estrinseca nel nome della variabile <varbuf$> che identifica il campo nel record (se è array l'indice non deve essere una variabile, ma costante, es. VA$(1)) e nella lunghezza in caratteri del campo <lungbuf> (massimo 255 byte) fino a un totale massimo di 128 campi per ciascun file. Se <lr> della OPEN è > della sommatoria delle varie<lungbuf> la differenza è assunta come tanti ASCII 0, diversamente l'esecuzione produrrebbe un errore.

I campi possono essere facoltativamente impaccati secondo i seguenti tipi di [tipoimp]:

- "Nessuno" per variabile stringa; se alla scrittura il contenuto del campo è< di <lungbuf> il campo è completato per la parte mancante con i caratteri ASCII 0; se invece è più lungo sono assunti i caratteri a partire da sinistra, con troncamento della parte eccedente;
- "ALFA" per variabile stringa; se alla scrittura e lettura il campo è automaticamente impaccato o disimpaccato: in lettura sono tolti tutti i caratteri ASCII 0 a destra; equivale ad usare la MKA$ prima della scrittura e la CVA$ dopo la lettura;
- "da NUM0 a NUM6" per variabile numerica; utilizzare da NUM0 a NUM6 in base ai decimali della variabile. Nella scrittura il campo viene trascritto in caratteri ASCII allineato a destra. L'eventuale virgola decimale viene scritta con il carattere punto "." La dimensione specificata in <lungbuf> determina il valore massimo del numero memorizzabile (ogni cifra occupa un carattere);
- "da BIN0 a BIN6" per variabile numerica; Utilizzare da BIN0 a BIN6 in base ai decimali della variabile. Il numero è scritto in binario; equivale a usare la MKN$ prima della scrittura e la CVN dopo la lettura; per i campi ad impaccamento binario "BIN" <lungbuf> deve essere un numero compreso tra 1 e 5 e determina il valore massimo del numero memorizzabile in base alla tabella specificata nella MKN$.

Se più di una FIELD fa riferimento allo stesso <numfile> sono considerate come un'unica FIELD di cui le successive, sono considerate come continuazioni di quelle precedenti.

# FIX

### FIX(<num>,<numdec>)

| num | numero da troncare |
|---|---|
| numdec | numero dei decimali al quale troncare |

In Sprix/Collage tronca i decimali del <num> al numero <numdec> di decimali (da 0 a 6) senza arrotondare. È automatica con <numedec>=6 sul risultato ottenuto e pertanto, in caso esecuzione di operazioni algebriche su valori con decimali può rilasciare risultati disattesi; questo automatismo è condizionato con SETDECFIX. Una chiamata esplicita di FIX ha prioritàsul FIX esegui to automaticamente o condizionato con SETDECFIX.

```
Esempio:
A=-12345.678: B=123.5: C=FIX(A,0): D=FIX(A,2): E=FIX(B,3):
risultati: C=-12345 : D=-12345,67 : E=123,5
```

# FOR NEXT

### FOR <varnum>=<x> TO <y> [STEP <z>] : '...' : NEXT

In Sprix/Collage esegue le istruzioni contenute in '...' un numero definito di volte stabilite dalla differenza fra il valore iniziale <x> e valore finale <y> del contatore numerico <varnum> (no array). Al termine di ogni ciclo, <x> si incrementa di <z> (default dell'incremento del contatore = 1 e può assumere anche valore <0) e il ciclo riprende dall'istruzione che inizia con la specifica FOR (se <x> <= <y>) altrimenti il programma prosegue alla prima istruzione successiva al comando NEXT (non modificabile, in quanto riferita all'ultima FOR aperta).

FOR-NEXT è nidificabile (una nell'altra fino ad 8) dove la NEXT della FOR interna precede la NEXT della FOR esterna.

```
Esempio:
FOR <varnum1>=<x> TO <y> [STEP <z>]
        ISTRUZIONE1...
FOR <varnum2>=<x> TO <k> [STEP <z>]
ISTRUZIONE2...
NEXT
NEXT
DIM C3(10,5)
FOR C1=1 TO 10
GOSUB UNO
        FOR C2=5 TO 1 STEP -1
GOSUB DUE : GOSUB TRE...
C3(C1,C2)=C1*C2: stampa C3(C1,C2)
NEXT
```

```
NEXT
END
```

*Nota: se il ciclo non è soddisfatto (ciclo null), l'istruzione viene eseguita almeno una volta. Esempio FOR I=1 TO LOF(n) con LOF(n)=0. È comunque possibile controllare il valore finale <y> per eseguire o saltare il ciclo.*

# GET

### GET <numfile>,<numrec>

In Sprix/Collage legge (GET) il numero di record <numrec> nel file aperto di tipo "R" numero <numfile>, utilizzando le variabili del buffer associato: in lettura sono riempite con i relativi valori letti (oppure tutte impostate a "" o 0 se il <numrec> è < 1 o > al numero di record presenti nel file).

# GETABBIN

### GETABBIN <codart$>,<codab$>,<nprog>,<cat$>

In Sprix legge (GET) i dati dei Codici Abbinati relativi alle variabili di struttura archivi _AB? in base al codice articolo <codart$> a cui sono abbinati (se "" = tutti gli articoli), al codice abbinato <codab$> (se "" = tutti gli abbinati), al numero progressivo per codici abbinati doppi <nprog> (se -1 = tutti, altrimenti specificare il numero di chiave che si intende prelevare: 0,1,2,..) e alla categoria di abbinamento <cat$> (se "" = tutte le categorie di abbinati).

La lettura rilascia _NAB cioè il numero di elementi dell'array dinamico della struttura _AB?

# GETAGP

### GETAGP [nrec],[dal$],[ala$],[cli$],[pre$],[ute$],[sta$],[ori$],[azi$]

In Sprix legge i dati delle righe della AGENDA DI LAVORO, relativi alle variabili di struttura archivi _AGP?, in base ai seguenti parametri identificativi obbligatori:

| | |
|---|---|
| **nrec** | Num. record (consente la ricerca diretta o la simulazione del ciclo) |
| **dal$** | Da data lavoro "AAAMMGG" |
| **ala$** | A data lavoro "AAAAMMGG" se vuota viene posta a "dal$" |
| **cli$** | Codice Cliente "GMM.CCCCC" |
| **pre$** | Codice prestazione |
| **ute$** | Utente "Nome di ricerca" |
| **sta$** | Stato Lavoro |
| **ori$** | Origine |
| **azi$** | Sigla azienda |

GETAGP ritorna un array di n. elementi contenuti nella variabile _AGPNELE.

# GETALIAS

### GETALIAS <codart$>,<codalis$>,<nprog>

In Sprix legge (GET) i dati dei Codici Alias relativi alle variabili di struttura archivi _AL? sulla base al codice articolo <codart$> a cui sono associati (se "" =tutti gli articoli), al codice alias <codalias$> (se "" = tutti gli alias) e al numero progressivo per i codici alias doppi <nprog> (se -1 = tutti, altrimenti specificare il numero di chiave che si intende prelevare: 0,1,2,..).

La lettura rilascia la _NAL per conoscere il numero di elementi dell'array dinamico della struttura _AL?

# GETAR

### GETAR <codart$>,<nummag>

In Sprix legge i dati di Anagrafica Articoli relativi alle variabili di struttura archivi _AR? in base al codice articolo principale <codart$> e gli eventuali progressivi del numero magazzino <nummag> (la lettura di magazzino = 0 significa totale magazzini ed è un dato di calcolo non memorizzato).

# GETAREA

### GETAREA <Area>

In Sprix legge i dati di Anagrafica Aree relativi alle variabili struttura archivi _AE?, in base al codice <Area>. La possibilità di cancellazione è attraverso il modulo Servizi di CdG.

# GETARVAR

### GETARVAR <codart$>,<dadata$>[,adata$]

In Sprix legge gli articoli e ne ritorna quelli variati a partire <dadata$>.

Valorizza _ERRARVAR=0 se l'operazione è corretta, altrimenti = 1 con messaggio d'errore riportato in _ERRARVAR$. Ritorna in variabile _CODARVAR$ (dal codice successivo a <codart$> se indicato, o dal primo se <codart$>=""), il

prossimo articolo variato dopo <dadata$> (e prima di [adata$] se impostata).

Notare: la conferma (con F10) della revisione di un articolo senza apportare alcuna modifica all'anagrafica, comporta comunque una riconferma e una variazione della data di ultimo aggiornamento. Questa variazione non avverrebbe invece con l'abbandono della revisione (con Esc).

```
Esempio:   Legge gli articoli variati dal 01/01/2011
 _CODARVAR$="" : DADATA$="20110101" : ADATA$="" : I=0
  LOOP:
 GETARVAR _CODARVAR$,DADATA$
  I=I+1
  LPRINT STR$(I,0,0)+") ("+_ERRARVAR$+"): "+_CODARVAR$
  IF _CODARVAR$<>"" THEN GOTO LOOP
```

Collage legge le _AR? con GETATTAR (senza i parametri): azzera le variabili preesistenti e carica i dati presenti a video fra l'evento START e END e fra PUT e DEL.

# GETASSOCDV

### GETASSOCDV [<tipo>,<keypass>,]<iddoc>

In Sprix legge i dati delle associazioni fra il Documento Docuvision e l'Entità del Gestionale, relativi alle variabili di struttura archivi _AD?, in base all'ID del documento Docuvision <idDoc> e il <tipo> di Entità del Gestionale, determinata univocamente dalla <keypass> di ricerca sulle tabelle d'associazione fra Gestionale-Docuvision. il <tipo> è obbligatorio se viene indicata la <keypass>.

Il parametro facoltativo [alleg] consente di variare la tipologia di allegato e il progressivo immagine attraverso la PUTASSOCDV. Il parametro ha la seguente sintassi "TPALL=x;PRGALL=y" dove "x" può assumere i seguenti valori null=Generico o I=Immagine o S=Scheda) o D=Descrizione e dove "y" può assumere valore numerico relativo al progressivo. Essendo parametro facoltativo,l'omissione di [alleg] equivale al default Generico "TPALL=;". Esempio di cambio di allegati Generici (associati ad "ART") in allegati immagine:

```
GETASSOCDV 1200,"ART",0
FOR X=1 TO _ADNUMASSOC
PUTASSOCDV 1200,"ART1",_ADDVDID(X),"TPALL=I;PRGALL="+STR$(X,0,0)
NEXT
```

GETASSOCDV valorizza _ADNUMASSOC che ritorna il num. globale d'associazioni.

La GET-PUT-DEL elaborano i dati _AD? in base ai seguenti parametri (eccetto per le classi <tipo>=1 Stampa generica, =800 Dichiarativi/Versamenti, =900 Stampe Fiscali; =1000 Agenda di lavoro ed eccetto i casi in cui il livello di azienda o il codice contratto non consentono l'accesso ad alcune tabelle,come es le operazioni di classe 'Parcella/Notulà senza gestione parcellazione, classi di Magazzino con azienda Contabile,...):

```
Piano dei Conti: <tipo>=400
  <keypass>="<codcon$>" (codice conto del piano dei conti)

Primanota: <tipo>=600
  <keypass> = "<dreg$>;<prinr>;<cau$>;<protr$>;<prots>;<protn>;<docn>;<ddoc$>"
                   <dreg$> (data di registrazione);
                   <prinr> (n. progressivo registrazione primanota);
                   <cau$>  (causale del documento);
                   <protr$>(registro di protocollazione)
                   <prots> (serie del protocollo)
                   <protn> (numero di protocollazione)
                   <docn>  (numero del documento)
                   <ddoc>  (data documento)

Anagrafica Articoli: <tipo>=1200
  <keypass> = "<codart$>" (codice articolo)

Movimenti di Magazzino: <tipo>=1400
  <keypass> = "<sig$>;<ser>;<num>;<clifor$>;<ddoc$>"
                   <sig$>    (sigla documento)
                   <ser>     (serie del documento)
                   <num>     (numero del documento)
                   <clifor$> (codice conto intestatario)
                   <ddoc$>   (data documento)

Ordine/Preventivi/Matrici: <tipo>=1500
  <keypass>="<sig$>;<ser>;<num>;<ddoc$>"
                   <sig$>  (sigla documento)
```

```
                    <ser>   (serie del documento)
                    <num>   (numero del documento)
                    <ddoc$> (data documento)

Bolla di Lavoro: <tipo>=1600
  <keypass> = "<numeroBL>;<numeroSottoBL>"
                    <numeroBL>      (numero di bolla di lavoro)
                    <numeroSottoBL> (numero di sottobolla di lavoro)
                    Inserire "-1" per avere tutte le sottobolle, inserire "0" (o non specificare) per intendere la BL principale. In
                    operazione DEL è richiesta anche la data della bolla.

Notule/Parcelle: <tipo>=1100
  <keypass>="<cau$>;<ser>;<num>;<clifor$>;<ddoc$>"
                    <cau$>    (causale del documento)
                    <ser>     (serie del documento)
                    <num>     (numero del documento)
                    <clifor$> (codice conto)
                    <ddoc>    (data documento)
```

# GETBL

### GETBL <numeroBL>,<numeroSottoBL>

In Sprix legge  i dati di Bolla di Lavorazione relativi alle variabili _OB? (testata) e _OL? (righe) in base al numero della bolla <numeroBL> e sottobolla <numeroSottoBL> di lavorazione (per le sottoBL dei lavoranti aggiungere 10000, es. la sottobl 0002->10002).

Il Collage legge le _OB? e _OL? con la GETATTBL (senza parametri): azzera le variabili preesistenti e carica dati presenti a video fra evento START e END e fra PUT e DEL.

La lettura valorizza _NBL, che in Sprix/Collage ritorna il numero delle righe in bolla lavorazione.

# GETCDG

### GETCDG <TipoDoc$>,<KeyDoc$>,<Anno>,<NumRigaDoc>,<NumProgAna>

Legge i dati dei movimenti di Contabilità Analitica relativi alle variabili _CDG?, in base a:

<TipoDoc$>    Tipo movimento (R=Primanota; M=Magazzino; C=Ord.Cliente; F=Ord.Fornitore; P=Preventivi; A=Marici; E=Extra)

<Anno>  nell'anno del movimento (contabile 1960 -2048 se tipo C/F, fisso 2048 se tipo E/P, fisso 1 se tipo A)

<NumRigaDoc>   al numero della riga del documento a cui sono associate le registrazioni di analitica (se 0=tutte le righe)

<NumProgAna>   e al numero progressivo di riga relativo all'analitico (se 0=tutte le righe; in magazzino è sempre 0 dato che per ciascuna riga di movimento è consentita una sola riga di analitica mentre in primanota ne sono consentite N).

<KeyDoc$>    La Chiave del movimento è formata da campi obbligatori di ricerca in base a <TipoDoc$> (separati da ;)

R - "PNDRE=x;PNNDO=x;" (PNNDO = num.progressivo registrazione primanota);

M - "MMSIG=x;MMNUM=x;MMSAZ=x;MMSER=x;MMCLI=x;NTMM=x;" (M è in relazione all'anno di ingresso; MMCLI è facoltativo per i documenti non intestati;

se NTMM - Num. riferimento testata documento - non è definito è impostato = 0);

C/F/P/A - "MMSIG=x;MMNUM=x;MMSAZ=X;MMSER=x;";

E - "PRGEXTRA=x;DTCOMP=x;AREA=x;COMM=x;CODANL=x;TIPMOV=x;" (il primo parametro PRGEXTRA non è obbligatorio);

La lettura rilascia la _CDGNR per conoscere il numero di elementi dell'array dinamico (numero righe) della struttura _CDG?

# GETCFR

### GETCFR <cod$>,<tipo>

Legge i riferimenti dei Referenti presenti nell'anagrafica della rubrica Clienti Fornitori (tasto funzione F4) relativi alle variabili di struttura _CFR? in relazione ai valori del Cliente/Fornitore <cod$> in base al <tipo> (se 0 = tutti i referenti, da 1 a 9 = un referente specifico). La lettura valorizza _CFRNELE (numero di referenti presenti); Il collegamento fra referente registrato e array Sprix è stabilito da _CFRPRG().

In Collage legge le variabili _CFR? con GETATTCFR (senza parametri); carica i dati presenti a video fra gli eventi ON_STARTCFR e ON_ENDCFR.

# GETCOM

### GETCOM <n>

In Sprix/Collage legge dalla porta seriale aperta la quantità dei dati <n> (da 1 a 255, con default 1 se < 1 o 255 se > 255) memorizzando i byte nella

# GETCOMM

### GETCOMM <Commessa>

In Sprix legge (GET) i dati di Anagrafica Commessa relativi alle variabili struttura archivi _CO?, in base al codice <Commessa>.

La possibilità di cancellazione è attraverso il modulo Servizi di CdG.

# GETCON

### GETCON <codice$>

In Sprix legge i dati di Anagrafica Contatti relativi alle variabili _DC? in base al codice contatto <codice$>

# GETDB

### GETDB <codart$>,<numfase>

In Sprix legge i dati della Distinta Base Primaria relativi alle variabili di struttura archivi _TP? in base al numero di fase <numfase> della DBP (1-99) e del codice articolo <codart$> con DBP.

Le operazioni sulla DBP valorizzano _NCOMP (numero dei componenti di una fase: max 5000 per fase).

La lettura valorizza il numero di fasi _NFASI della DBP. Nella lettura della grande DBP elaborare fase per fase (una alla volta, senza definire 49500 potenziali variabili). La scrittura è per numero di fase su articolo esistente (_TPCOD$) con DBP a una/più fasi.

# GETDVDOC

### GETDVDOC <numdoc>,<numrev>,[dirloc]

In Sprix legge i dati del Documento Docuvision relativi alle variabili struttura archivi _DV?, in base al numero Documento di Docuvision <numdoc> e numero revisione <numrev> (in lettura se -1=tutte le revisioni, 0=ultima revisione, >0=revisione specificata;in cancellazione se 0=cancella l'intero documento).

Facoltativa è la possibilità di copiare i documenti allegati in una directory locale, attraverso il parametro [dirloc] e la sintassi specifica per il parametro è "DIRLOC=". Per esempio "DIRLOC=<C:\TEMP\dv\". Notare il simbolo"<" che indica la directory locale e la barra finale della directory scelta.

Le operazioni verificano i Parametri Generali di Docuvision poichè la gestione delle pagine è alternativa a quella delle revisioni di documenti già con la gestione pagine attiva è gestita solo l'ultima <numrev> contenuta in<numdoc> e quindi viene ignorata sia in lettura (i dati sono riferiti all'ultima revisione) che in cancellazione (quindi va valorizzata = 0).

La lettura valorizza _DVDNUMVER (in scrittura è calcolata) e _DVDPAGXVER(), che in Sprix ritornano il numero globale di revisione letta (=1 con gestione pagine attiva) ed il numero di pagine per revisione ( =1 con gestione pagine non attiva) che permette di scorrere i vettori delle variabili contenenti i dati delle revisioni (tutti uguali eccetto il contatore di pagina) e il vettore dei nomi file _DVVFILEGET.

L'elaborazione è solo per l'azienda aperta: non sono gestiti i documenti di altre aziende eccetto quelli senza sigla (utilizzabili da tutte le aziende);la scrittura verifica che l'azienda specificata sia quella dell'azienda aperta (diversamente il doc. è disponibile per tutte le aziende) e la scrittura del documento con sigla diversa da quella aperta non è consentita (la sigla azienda documento è automatica con quella dell'azienda aperta). L'inserimento di nuove pagine o la modifica di una revisione esistente, richiede variabili correttamente impostate: la scrittura non avviene se non si associa almeno una corretta revisione in 'senso latò (cioè anche se è il 1^ inserimento e senza che siano mai avvenute revisioni in 'senso strettò). Notare: ulteriori informazioni sulle revisioni sono in calce alle variabili _DV?

# GETESPOS

### GETESPOS <cod$>,<data$>

Funzione Sprix/Collage di lettura esposizione alla data <data$> (formato "AAAAMMGG") relative al conto <cod$> (formato "GMM.CCCCC"). Ritorna eventuali errori nella variabile _ESPOSIZERR$:

```
    -   "codice di tipo diverso da C/F"
    -   "codice non trovato in pico" (piano dei conti)
    -   "codice non trovato in rudt" (rubrica)
    -   "data per calcolo esposizione non gestita/errata"
```

Nota bene: la funzione lavora sull'anno aperto (modificabile con APRIAZIE)

La funzione rilascia informazioni nelle seguenti variabili:

| _SALDOCLIFOR | - saldo contabile del cliente/fornitore |
|---|---|
| _FIDOCLIFOR | - valore fido (anche calcolato su fatturato anno precedente) |
| _VALORDBOLLE | - valore ordini/bolle |
| _VALORDINI | - valore ordini netto |
| _VALORDINII | - valore ordini con IVA |
| _VALBOLLE | - valore bolle netto |
| _VALBOLLEI | - valore bolle con IVA |
| _ESPOSIZIONE | - esposizione bancaria |
| _DATAESPOS$ | - data a cui e calcolata l'esposizione (DATA parametro) |
| _VALRISCHIO | - valore di rischio, come da formula della tabella progressivi |
| _FUORIFIDO | - differenza fra FIDOCLIFOR e VALRISCHIO (solo se rischio>fido) |

# GETIL

### GETIL ndoc, nsot, nropf, fase, mag, cod

| ndoc | Numero della bolla di lavorazione, corrisponde alla variabile _ILNDO |
|---|---|
| nsot | Numero della sotto bolla di lavorazione, corrisponde alla variabile _ILNSD |
| nropf | Numero riferimento prodotto finito, corrisponde alla variabile _ILNRB |
| fase | Numero di fase del componente, corrisponde alla variabile _ILFAS |
| mag | Magazzino materie prime, corrisponde alla variabile _ILMAG |
| cod | Codice del componente, corrisponde alla variabile _ILAMP$ |

In Sprix legge i dati degli Impegni di Lavorazione relativi alle variabili di struttura archivi _IL? in base al numero BL <ndoc> e num.sottobolla <nsot>; la specificazione del codice della materia prima <cod> (se "" = tutte le MP) implica l'indicazione del numero di magazzino <mag> (se 0 = tutti i magazzini) che, se specificato a sua volta implica il numero fase <fase> (se 0 = tutte le fasi) che, se specificata implica il num. di rifer. prod. finito <nropf> (se 0=tutte, se -1=RES_SL,se -2=RES_DL; attenzione, non essendo gestiti i numeri negativi, sostituire per RES_SL il valore 65534 al posto di -1 e per RES_DL 65533 al posto di -2.)

La lettura valorizza la _NIL che in Sprix ritorna il num. righe di componenti Materia Prima.

# GETIND

### GETIND <codice$>, [codice_conto$]

In Sprix legge i dati di Anagrafica Indirizzi di Spedizione relativi alle variabili di struttura archivi _DA? in base al codice indirizzo di spedizione <codice$> e all'eventuale [codice_conto$] cui è associato l'indirizzo ("" se non utilizzato).

# GETINTERNET

### GETINTERNET <url$>,<file$>

In Sprix/Collage e sistemi Windows/Linux copia un file con il nome <file$> estensione ".html" dall'indirizzo <url$> di un sito Internet sul proprio computer locale e nel percorso indicato nello stesso nome del <file$>. La porta di default è 80, se quella del Server WEB è diversa occorre iserire il prefisso "[porta:nnn]" in <url$>, dove nnn è il numero della nuova porta.

GETINTERNET valorizza _ERRGETINT$ che riporta la descrizione dell'eventuale errore.

# GETKEY

### GETKEY <numfile>,<numkey>,<key$>[,key1$]

In Sprix/Collage cerca la chiave stringa <key$> nel file a indici aperto con numero <numfile>, avvalendosi della chiave numero <numkey> (da 1 a 4, cioè una fra quella primaria e le secondarie). Le variabili del buffer associato vengono riempite con i corrispondenti valori della chiave trovata; se non è trovata, le variabili assumono i valori della chiave immediatamente successiva e se anche questa non esiste tutte le variabili del buffer sono impostate a "" oppure 0. [key1$] è parametro facoltativo con la chiave principale in caso di ricerca per chiave secondaria di più record con la stessa chiave secondaria (individua il record cercato anzichè leggere quello con la chiave principale più piccola). Valorizza _NRDATI e _ERRIND.

# GETLOT

### GETLOT <idlot>

In Sprix legge i dati di Anagrafica Lotto relativi alle variabili struttura archivi _LOT? in base al numero ID lotto <idlot>. La lettura valorizza _NDULOT (numero dati utente gestiti dal lotto). Se il lotto è presente in archivio sono assunti i dati in archivio: il Tipo controllo carico; Fornitore/Data carico (se lotto caricato da movimento e tipo controllo carico 'F'/'D'); dati utente gestiti dal lotto e non nelle variabili Sprix. Se il lotto invece è nuovo sono assunti i dati Sprix o dal default del tipo lotto associato: i dati utente non impostati; il carico movimento magazzino e il tipo controllo carico (non valorizzabili con Sprix);

codice articolo che gestisca dei progressivi (se indicato uno che non li gestisce); codice utente lotto viene impostato con i dati personalizzati. A parità di dati aggiuntivi, la variazione dei soli dati generali comporta la sola scrittura di questi ultimi.

# GETMM

### GETMM <doc$>

In Sprix legge (GET) i dati del Documento di Magazzino relativi alle variabili struttura archivi _MM? in base al documento <doc$> in forma SIGLA[SERIE/]NUMERO (o SIGLA/NUMERO+SIGLA/NUMERO+ecc se più di uno); la sintassi identifica documenti dell'anno solare dell'azienda aperta (per l'infrannuale, il parametro di magazzino "Numerazione solare mov.mag(S/N): N" consente di inserire l'anno del documento -nn- prima del protocollo SIGLASERIE/nn-NUMERO, es. GETMM "FT5/08-3432" - per la infrannuale aperta nel 2009 - o GETMM "FT5/09-3432" - per quella aperta nel 2008 -).

Il Collage legge le _MM? con GETATTMM (senza parametri): azzera le variabili preesistenti e carica dati presenti a video fra evento START e END e fra PUTe DEL (lo START si attiva dopo l'uscita dal primo campo che divide l'emissione dalla revisione, mentre in emissione differita si attiva con trasformazione del documento e non con la ristampa).

La lettura valorizza _NMM e _NTMM, che in Sprix/Collage ritornano il numero di righe documento di magazzino (in 'R' elementi) e il numero di testate documenti origine/trasformati (in 'T' elementi) che lo compongono (se non azzerate da AZZVARSYS azzerarle direttamente inizializzandole = a zero).

L'identificazione del documento presente in archivio più di una volta ma di clienti/fornitori diversi apre una finestra di selezione del documento (e in cui, se premuto [Esc], sono azzerate alcune variabili es. _ERRMM$, _NMM):

# GETMM_EXT

### GETMM_EXT <doc$>[,conto$]

Evita l'apertura della finestra consentendo di specificare il codice del conto del cli/for [,conto$] per identificare univocamente il documento (se non viene indicato si comporta come GETMM);

# GETMULTILOT

### GETMULTILOT<[TipoLotto],[Codart]>,[TipoCodUt],[CodUtente],<Includi>,<Mag>

In rosso i dati obbligatori

| | |
|---|---|
| TipoLotto | Tipo del lotto obbligatorio in alternativa al codice dell'articolo |
| Codart | Codice dell'articolo proprietario dei lotti, obbligatorio in alternativa al tipo lotto |
| TipoCodUt | = 0 completo<br>= 1 per la parte iniziale<br>= 2 se parziale |
| CodUtente | Codice utente del lotto |
| Includi | 1 = tutti<br>0 = tutti tranne quelli con statoB/N, con data validità inferiore alla data terminale e quantità 0 |
| Mag | Numero del magazzino |

In Sprix legge e carica l'elenco dei lotti relativi al codice di tipo lotto [TipoLotto] o al codice articolo [Codart],.uno fra i due è obbligatorio, se inseriti entrambi in ricerca prevale Codart, nelle variabili array di struttura archivi _LOT?_M(), il cui numero di elementi è rappresentato dalla variabile _LOTNID_M e cioè il numero di ID letti e impostati da GETMULTILOT.

Se è impostato il codice utente [CodUtente] per ritornarne i suoi lotti il [TipoCodUt] stabilisce se questo codice utente sia completo (tipo=0), per la parte iniziale (tipo=1) o se parziale (tipo=2); obbligatorio <Includi> se=1include tutti, come F3; se=0, come F2, include tutti tranne quelli con statoB/N, con data validità inferiore alla data terminale e quantità 0); obbligatorio il numero di magazzino <Mag> (numero 1-255; se = 0 significa tutti i magazzini, non operativo se <Includi> = 1).

# GETNUM

### GETNUM (<Tiposigla$>,<SiglaDoc$>,<Anno>,<Serie>)

In Sprix ritorna l'ultimo numeratore e setta il numeratore (SET) relativo all'area <Tiposigla$> ("C"=contabile,"M"=magazzino) del <SiglaDoc$> (tipo del sezionale A/V/S/C/R/X/F se l'area è "C" o la sigla del documento di magazzino se l'area è "M") riferito all'anno solare <anno> e al numero di <Serie> (1-99). Valorizzano l'esito operazione in _ERRNUM$

# GETNUMSTRUT

### GETNUMSTRUT <codart$>

In Sprix/Collage ritorna un valore numerico utile a identificare la struttura dell'articolo con codice principale <codart$>. I valori sono i seguenti:

1-18 se <codart$> ha radice minima codificata e collegata a una struttura;
0 se <codart$> ha radice minima codificata anche se non collegata ad alcuna struttura;
-1 se <codart$> non ha radice minima codificata

GETNUMSTRUT è utilizzabile per sapere se la radice minima di <codart$> può essere inserita oppure no. Se invece l'esigenza è quella di sapere se un codice esiste oppure no, basta utilizzare il dizionario ardes("codart$") che, se ritorna valore significa che codart$ esiste, altrimenti, se non ritorna alcun valore significa che codart$ non esiste.

Esempio (ipotesi: solo due codici nell'installazione)
PF1 appartenente alla struttura 1
PF2 non appartenente a nessuna struttura
GETNUMSTRUT di qualunque <codart$> che inizia con PF1 ritorna valore = 1
GETNUMSTRUT di qualunque <codart$> che inizia con PF2 ritorna valore = 0 sia che <codart$> esista oppure no.
GETNUMSTRUT di qualunque <codart$> che inizia con valori diversi da PF1 o PF2 ritorna valore = -1

# GETPART

### GETPART <tppart$>,<codcon$>,<catcon>,<codart$>,<catart>,<dtvalda$>

In Sprix legge i dati delle Particolarità Conti e Articoli relativi alle variabili struttura archivi _PART?,in base al tipo particolarità <tppart$>(P,S,V,A), al codice conto <codcon$>(o in alternativa al suo numero di categoria <catcon> listino, sconto, provvigione), al codice articolo <codart$> (o in alternativa al suo numero di categoria <catart> formazione prezzo, sconto, provvigione) e con data d'inizio validità <dtvalda$> della particolarità. La combinazione di quattro parametri individua la specifica particolarità e di questi, il secondo e il terzo sono combinabili in quattro modi:(<codcon$>,<codart$>); (<codcon$>,<catart>); (<catcon>,<codart$>); (<catcon>,<catart>).

Collage legge anche con GETATTPART (senza parametri): azzera le variabili preesistenti e carica i dati presenti a video fra evento START e END e fra PUT e DEL.

# GETPARTICOL

### GETPARTICOL <tppart$>,<codcon$>,<catcon>,<codart$>,<catart>,<dtvalda$>,[mod]

In Sprix/Collage attiva la modalità di inizio ricerca delle particolarità che soddisfano i requisiti dichiarati nei parametri (se [mod]=0 ricerca dalla 1^ paritolarità congruente ai requisiti, se [mod]<>0 ricerca le successive congruenti).

<dtvalda$> non viene controllato;

se cambiano i tipi di parametri, impostare nuovamente GETPARTICOL con dei nuovi parametri per leggere le ulteriori particolarità.

Collage legge anche con GETATTPART (senza parametri): azzera le variabili preesistenti e carica i dati presenti a video fra evento START e END e fra PUT e DEL.

# GETPC

### GETPC <codcon$>

In Sprix scrive (PUT) legge (GET) cancella (DEL) i dati del Piano dei Conti relativi alle variabili di struttura _PC? in base al codice conto <codcon$>.

La PUTPC del conto nuovo richiede i dati obbligatori della versione del gestionale in uso; la PUTPC di un conto già esistente lo aggiorna (valorizzare sempre tutte le variabili di interesse, altrimenti alcune potrebbero azzerarsi). Prestare anche attenzione alle informazioni relative al caricamento dei conti tramite la trasmissione CSV.

ATTENZIONE: in ambiente a file separati (tecnologia "BP"), per le aziende di gruppo, la PUTPC non è fruibile sui conti diversi da quelli di tipo C/F/B, perchè gli unici conti che questa processa sono i clienti, fornitori e banche (cioè quelli propri dell'azienda in questione). Tramite la variabile _PCSOLOPICO$ = "S/N" (con default = "N") si possono aggiornare tutte le variabili tipiche del conto aziendale anche per i conti di raggruppamento.

I dati della rubrica (RUDT) rimangono invariati. Per i SOLI conti presenti nell'azienda le variabili che possono essere modificate quando la variabile _PCSOLOPICO$ è impostata = "S", sono contrassegnate con il simbolo (*a) nella tabella di variabili _PC? elencate alla fine del paragrafo.

# GETPN

### GETPN <dreg$>,<prinr>,<cau$>,<protr$>,<prots>,<protn>,<docn>,<ddoc$>

In Sprix legge i dati della registrazione di Primanota relativi alle variabili di struttura archivi _PN?, in base alla data di registrazione <dreg$>, il numero progressivo di registrazione di primanota <prinr> (1-16.777.214), causale del documento <cau$>, registro di protocollo <protr$>, serie di protocollo <prots>, numero della protocollazione <protn>, numero del documento <docn>, data del documento <ddoc$>. La <dreg$>è obbligatoria e con <prinr> forma la chiave primaria di

Primanota: l'omissione di <prinr> (se = 0) rende i restanti parametri necessari per identificare e discriminare le registrazioni con dati equivalenti (altrimenti il documento processato è il primo che viene trovato in archivio) e pertanto, devono assumere valore diverso da 0 (se numerico) o "" (se stringa); l'indicazione di alcuni parametri = 0 o = "" può infatti identificare più di un documento, fra i quali il primo incontrato è processato.

Collage legge le variabili _PN? con la GETATTPN (senza parametri): azzera le variabili preesistenti e carica i dati presenti a video fra l'evento START e END e fra la PUT (anche proveniente da registrazioni di magazzino) e la DEL.

Il progressivo <prinr> è contenuto nella _PNPRN di sola lettura e le GET in Sprix/Collage valorizzano _NPN (numero di righe lette nel documento).

# GETPOSCURSX

### GETPOSCURSX(<par>)

Ritorna la colonna in cui e' posizionato il cursore in quel momento. <par> assume valore 1 se la finestra e' di Sprix/Collage oppure 2 se la finestra è del gestionale nella routine Collage, indipendentemente dall'interfaccia. L'utilizzo della ATTIVAFIN riposiziona alle coordinate 1,1 della finestra. Unica eccezione le finestre elenco articoli e Clienit/fornitori dove il valore ritorna impostato al massimo delle colonne gestite.

# GETPOSCURSY

### GETPOSCURSY(<par>)

Ritorna la riga in cui e' posizionato il cursore in quel momento. <par> assume valore 1 se la finestra e' di Sprix/Collage oppure 2 se la finestra è del gestionale nella routine Collage, indipendentemente dall'interfaccia. L'utilizzo della ATTIVAFIN riposiziona alle coordinate 1,1 della finestra.

# GETRUBUNICA

### GETRUBUNICA <id>

In Sprix legge i dati di Rubrica Unica per l'<id> numerico specificato.

Il parametro <id> è l'identificativo della Rubrica Unica. Valorizza le variabili di struttura _PC??? dei soli dati di Rubrica. Valorizza le variabili __ERRUBUNI e __ERRUBUNI$.

Collage legge le variabili _PC? con GETATTPC (senza parametri): azzera le variabili preesistenti e carica i dati presenti a video fra l'evento START ed END e fra PUT e DEL.

# GETTPLOT

### GETTPLOT <tplot$>, [ord]

In Sprix legge (GET) i dati di Anagrafica Tipo Lotto relativi alle variabili struttura archivi _TPLOT? in base al codice Tipo Lotto <tplot$> (in caso di nuovo tipo lotto indicare "**") e l'eventuale ordine [ord] di lettura dati utente personalizzati (1=per codice dato; 2=per come immessi a video -default se non viene indicato-; 3 = in base all'ordine impostato per il lotto). La Lettura valorizza _NDUTPLOT (numero dati utente gestiti).

# GIORSET

### GIORSET(<data$>)

In Sprix/Collage ritorna il giorno della settimana della stringa <data$> in formato "AAAAMMGG" compreso fra 19600101 e 20481231. I numeri relativi ai giorni sono: 1 (lun), 2 (mar), 3 (mer), 4 (gio), 5 (ven), 6 (sab), 7 (dom)

```
Esempio
OGGI$="20090618":  N=GIORSET(OGGI$)  risultato  N=4  (giovedi)
```

# GOTO

### GOTO <nometic>

In Sprix/Collage salta alla subroutine dell'etichetta <nometic>.

```
Esempio:
X=0
INIZIO:
X=X+1: visualizza X:
IF X=3 THEN GOTO FINE ELSE GOTO INIZIO
```

```
FINE:
END
```

# GOSUB

### GOSUB <nometic> ... RETURN

In Sprix/Collage salta alla subroutine dell'etichetta <nometic> dello stesso programma, terminata la quale ne esce con RETURN per tornare al punto di partenza e successivo all'ultimo GOSUB eseguito. GOSUB è nidificabile (uno nell'altra fino a 30). In caso di ciclotabellare, si esce con EXITTAB e non con RETURN (da posizionare dopo e fuori il ciclotabellare).

```
VA$="var_alfa": FOR I=1 TO 3 .... NEXT
IF A=1 THEN N=10: GOSUB ETIC1 ELSE …
 GOTO FINE
ETIC1:
RETURN
FINE:
END
```

# INPUT

### INPUT <tipo$>,<str$>,<larg>

| | |
|---|---|
| tipo$ | “A” Alfanumerio<br>“AM” Alfanumerico maiuscolo<br>“AE” Alfanumerico caratteri estesi comprese le accentate<br>“D” Data accetta date dal 01/01/1960 al 31/12/2048<br>“DL” data libera accetta qualunque data<br>“C” Conti del piano dei conti<br>"C+" solo conti Clienti e Fornitori<br>"C-" tutti i conti, escluso i Clienti e Fornitori<br>"C?" ? identifica il tipo di conto ammesso: M=Merci, S=Spese, V=Vendite beni amm., L=Costi pluriennali, P=Patrimoniali generico, C=Clienti, F=Fornitori, I=Immobilizzazioni, K=Costi pluriennali, R=Rimanenze, O=Conti d'ordine, Z=Conti sospesi). Esempio "CC".<br>“M” Articolo di magazzino<br>“N(1-6)” Numerico fino a un massimo di 6 decimali |
| str$ | Varibile dichiarata per l’input |
| larg | Dimensione visbile del campo d’input |

In Sprix/Collage consente l'immissione di <str$> (numerico o stringa) da tastiera nella finestra attiva, alla posizione attuale del cursore, in un campodi massimo <larg> caratteri in base al <tipo$> di immissione, cioè: N, N1, N2, N3, N4, N5, N6 (numero con 0,1,2,3,4,5,6 decimali), A (alfanumerico), AM(alfanumerico maiuscolo), AE (caratteri estesi es. lettere accentate), C o M(Conto Contabile o Articolo di Magazzino), DL (data libera < 01/01/1960 o > 31-12-2048), D (data e controllo di validità). Attiva le funzioni monitor del gestionale (ricerca, modifica, inserimento...). Vedere anche PARAMETRI DI ESECUZIONE [F6].

Se <tipo$> = "D", <larg> deve essere 10 o 8 o 5 se <str$> di input è nella forma GG/MM/AAAA o GG/MM/AA o GG/MM (con AAAA assunto in data odierna); la variabile è sempre impostata nella forma "AAAAMMGG".

Se <tipo$> = "M" o "C" la <larg> superiore alla dimensione del codice visualizza la descrizione nella parte eccedente; la variabile viene impostata con il codice.

Per differenziare i conti è consentito il seguente parametro <tipo$>:

"C" (tutti i conti)
"C+" (solo conti Clienti e Fornitori)
"C-" (tutti i conti, escluso i Clienti e Fornitori)
"C?" (dove ? identifica il tipo di conto ammesso: M=Merci,
S=Spese, V=Vendite beni amm., L=Costi pluriennali,
P=Patrimoniali generico, C=Clienti, F=Fornitori,
I=Immobilizzazioni, K=Costi pluriennali, R=Rimanenze,
O=Conti d'ordine, Z=Conti sospesi). Esempio "CC".

*ATTENZIONE: questa differenziazione dell'istruzione NON si estende anche ai PARAMETRI DI ESECUZIONE [F6].*

Se <str$> contiene già un valore questo è proposto come dato di ingresso modificabile (viene visualizzato il contenuto della variabile, il cursore si ferma sul primo carattere del campo, accetta e controlla l'immissione/variazione del campo stesso); se è stringa e contiene un carattere ASCII 0, questo è considerato come la fine della stringa; se è array l'indice deve essere una costante, es. INPUT "A",VAR$(2),1

INPUT è condizionata dalla _SCORRI e valorizza la _TF (eccetto la pressione dei tasti funzione monitor quando <tipo$>="C", "M" o "D" (non c'è un'uscita dal campo e non ci sono eventi); infine, non consente la scelta del font.

Esempio
```
D$="": N$="SCRIVI IL TUO NOME": LOCATE 5,1: INPUT "AM",N$,30
LOCATE 6,1: PRINT "allora ti chiami "+N$+" . Data di nascita ? ";:
E$="D": INPUT E$,D$,10
```

# INPUT_EXT

### INPUT_EXT <numero zonamouse>,<tipo$>,<str$>,<larg>,<maxlunstr>,<param$>

Istruzione che abilita input a scorrimento.

| <numero zonamouse> | vale da 1-127 (0 = disabilita la zona mouse; -1 = disabilita tutte le zone mouse della finestra); |
|---|---|
| <param$> | ha valore 'null' (quindi va scritto come "") |
| <tipo$>: | vedere istruzione INPUT. Solo per INPUT_EXT<br>"F" tipo file abilita la navigazione del file system del server o del client se indicato il simbolo di reindirizzamento "<", nella variabile <str$> ritorna il percorso con il nome file<br>"P" tipo direcory abilita la navigazione delle directory del file sistem del server o del client se indicato il simbolo di reindirizzamento "<", nella variabile <str$> ritorna il percoso selezionato |
| <str$> | vedere istruzione INPUT |
| <larg> | Indica il numero dei caratteri visbili dell'input |
| <maxlunstr> | Dimensione massima del campo di input |
| <param$> | Parametri della funzione<br>Tipo = F<br>NOCANC disabilita la cancellazione dei file<br>NOESEC disabilita la possibilità di eseguire i file<br>Tipo = P<br>CREADIR abilita la creazione delle directory. |

Variabili gestite:
_TMZ -> ritorna il numero della zonamouse;
_TMR -> ritorna il numero riga della finestra;
_TMC -> ritorna il numero colonna della finestra;

Nel seguente esempio, cliccando una qualsiasi area della finestra sono riportate le coordinate di riga/colonna (TMR/TMC) in cui è avvenuto il "click".

Cliccando sulla zonamouse relativa alla PRINT "XXXXXXXXXX" viene rilasciato il codice TMZ=111, mentre su zonamouse della INPUT_EXT il codice TMZ è 99.

```
APRIFIN 1,1,0,0,0,10,50
COLOR 3: LOCATE 2,10:PRINT "XXXXXXXXXX": COLOR 0
SETFSM 1
SETZM 2,10,10,111:
VIS:
LOCATE 3,1: INPUT_EXT 99,"A",A$,20,100,""
LOCATE 4,1: PRINT "TF :   ": LOCATE 4,5: PRINT _TF
LOCATE 5,1: PRINT "TMZ:   ": LOCATE 5,5: PRINT _TMZ
LOCATE 6,1: PRINT "TMR:   ": LOCATE 6,5: PRINT _TMR
LOCATE 7,1: PRINT "TMC:   ": LOCATE 7,5: PRINT _TMC
GOTO VIS:
```

Esempio di input di tipo File

INPUT_EXT 0,"F",MIAVAR$,30,128,"" input di tipo file con tutte le opzioni
INPUT_EXT 0,"F",MIAVAR$,30,128,"NOCANC" input di tipo file spegne la cancellazione dei file
INPUT_EXT 0,"F",MIAVAR$,30,128,"NOESEC" input di tipo file spegne l'esecuzione dei file
INPUT_EXT 0,"F",MIAVAR$,30,128,"NOESEC;NOCANC" input di tipo file spegne l'esecuzione e la cancellazione dei file

**Apertura automatica di un file:**
```
NOME_FILE$ = "<C:\Appoggio\Notule_farmacie.pdf"
PUTTAST "#605#501#501"
INPUT_EXT 0,"F",NOME_FILE$,1,128,","
```

Esempio di input di tipo Directory

INPUT_EXT 0,"P",MIAVAR$,30,128,"" input di tipo directory con tutte le opzioni
INPUT_EXT 0,"P",MIAVAR$,30,128,"CREADIR" input di tipo directory con abilitazione di creazione direcotry

# INSKEY

### INSKEY <numfile>

In Sprix/Collage nel file a indici aperto con numero <numfile> inserisce un record con i valori contenuti nelle variabili del buffer associato. In inserimento: se i valori delle variabili sono più lunghi dei buffer è scritta solo la parte non eccedente, diversamente, se inferiori, sono utilizzati ASCII 0 per l'eccedente l'inserimento incrementa a multipli di 64 record e i record non utilizzati contengono tutti valore ASCII 0; l'inserimento non è eseguito; se la chiave esiste; tutte le chiavi sono obbligatorie e i loro valori non possono essere tutti ASCII 0 o ASCII 255. Se i campi contenenti chiavi sono variati i relativi file chiavi vengono riordinati. Più chiavi secondarie uguali del nuovo record vengono inserite in ordine di principale + secondaria.

Fare attenzione a tutto il buffer (la dimensione della chiave deve mantenere la lunghezza). Valorizzano _NRDATI e _ERRIND.

# INSTR

### INSTR(<str$>,<ricercastr$>)

In Sprix/Collage cerca la stringa <ricercastr$> nella stringa <str$> ritornandone la prima posizione in cui la trova (se non c'è ritorna 0).

Esempio
A$="spRix":B$="Ri" il risultato di INSTR(A$,B$) o INSTR(A$,"Ri") è 3

# INSTR_CI

### INSTR_CI(<str$>,<ricstr$>)

In Sprix/Collage cerca in modalità CASE INSENSITIVE della stringa <ricstr$>nella stringa <str$> ritornandone la prima posizione in cui la trova (se non c'è ritorna 0). Non distingue fra caratteri minuscoli e maiuscoli.

Esempio
A$="spRix":B$="rI" il risultato di INSTR_CI(A$,B$) o INSTR_CI(A$,"rI") è 3

# INKEY

### INKEY()

In Sprix/Collage controlla la pressione del tasto da tastiera; ritorna zero in caso negativo oppure il valore ASCII (o del codice inserito in SETTF) del tasto premuto. L'utilizzo sconsiderato di INKEY impegna molto le risorse del sistema, soprattutto se accorpa altre istruzioni/funzioni pesanti, pertanto valutare l'opportunità di gestire SLEEP e _INTER=0 ( _INTER=1 può disattivare la INKEY ).

```
_INTER=0: PRINT "Premi F5..."
 LOOP: SLEEP 100: A=INKEY(): IF A=0 THEN GOTO LOOP ELSE PRINT "Premuto..";:
 IF A=605 THEN PRINT "F5": INPUT "N",N,1: _INTER=1
```

# KILL

### KILL <nomefile$>

In Sprix/Collage cancella il file dal nome <nomefile$> che deve esistere, essere chiuso e può essere eventualmente preceduto dalla directory.

# LEFT

### LEFT$(<str$>,<numcar>)

In Sprix/Collage estrae i primi caratteri <numcar> a partire da sinistra della stringa <str$> (se vuota ritorna ""); se la lunghezza di <numcar> è > di<str$> ritorna l'intera <str$>. Es. A$="Sprix": B$=LEFT$(A$,3) ris. B$="Spr"

Esempio:
C$="SPRIX": D$=LEFT$(C$+STRING$(10," "),10): risultato D$="SPRIX     "

# LEGGIDESC

### LEGGIDESC <par>

| | |
|---|---|
| par | 1 = abilitato Vengono lette le righe descrittive del ciclo tabellare<br>0 = disabilitato Non vengono lette le righe descrittive del ciclo tabellare |

In Sprix se <par>=1 sono abilitate, il default è <par>=0 cioè disabilitate. Il comando deve essere impostato prima della definizione del ciclo tabellare.

# LEGGIDIR

### LEGGIDIR <directory$>,[maschera$]

In Sprix/Collage ritorna il contenuto (elenco file) presenti nel percorso <directory$>, con eventuale filtro di ricerca file [maschera$] (es."*.txt").

Valorizza _ERRDIR$ (eventuale errore su directory), _NUMDIRFILE (num.di file nella directory), _DIRFILE$(I) (array stringa coi nomi file nella directory)

# LEN

### LEN(<str$>)

In Sprix/Collage ritorna la lunghezza della stringa <str$> (se vuota o non esiste è 0)

Esempio:
```
A$=" A 1 B " il risultato di LEN(A$) o LEN(" A 1 B ") è 7
```

# LOCATE

### LOCATE <rig>,<col>

In Sprix/Collage posiziona il cursore alle coordinate di riga <rig> e colonna <col> nella finestra attiva. Nella finestra base (0) la <rig> va da 1->20(riga fisica 4->23) e la <col> va da 1->78/94 (colonna fisica 2->79); nelle altre finestre (1-9) la<rig>/<col> vanno da 1 alla lunghezza/larghezza della finestra -2.

Esempio:
```
APRIFIN 1,1,0,3,7,9,12 : LOCATE 1,2
```

# LOF

### LOF(<numfile>)

In Sprix/Collage ritorna la lunghezza (quantità caratteri) del file numero <numfile>. Ritorna 0 se il file è vuoto, ritorna -1 se non è stato aperto.

# LOFKEY

### LOFKEY(<numfile>)

In Sprix/Collage ritorna la quantità di chiavi (quindi di records) inserite nel file a indici numero <numfile>. Ritorna 0 se il file è vuoto, ritorna -1 se non è stato aperto o non è a indici.

# LPRINT

### LPRINT <espr$>[;]

In Sprix/Collage invia l'espressione stringa <espr$> alla stampante aperta.

Se assente [;], dopo la stampa di <espr$> è effettuato un salto riga. Le caratteristiche di stampa dipendono dalla configurazione, impostazione e caratteri del dispositivo hardware. Se utilizzata in concomitanza (prima o dopo) di altre istruzioni di stampa (es. barcode/immagine) dotate di parametri di posizionamento, si possono generare sovrapposizioni (poichè indipendenti).

Esempio:
```
A$="Spx_1":B$="Col_2":C$="Sdk_4": LPRINT A$+B$+"Mds_3";: LPRINT C$
INIZCO$()
FINECO$()
INIZES$()
FINEES$()
INIZGR$()
FINEGR$()
INIZIT$()
FINEIT$()
INIZSO$()
FINESO$()
```

In Sprix/Collage ritornano una stringa di caratteri da inviare alla stampante per impostare l'INIZIO e la FINE del modo di stampa in: compresso (CO$), espanso (ES$), grassetto (GR$), corsivo (IT$), sottolineato (SO$). Non tuttii tipi di formato consentono interventi sullo stile del font (es. sottolineato in pdf, corsivo o espanso in rtf,...).

Esempio

```
LPRINT INIZCO$(); LPRINT "inizio riga in compresso": LPRINT FINECO$();
```

## LPRINTBARCODE

### LPRINTBARCODE <codice$>,<tipo>,<prncod>,<um>,<x>,<y>,<cx>,<cy>

In Sprix/Collage stampa su stampanti configurate di tipo 'W' o 'F' la stringa codice a barre <codice$> (con ultimo carattere 'check digit' corretto) di tipo <tipo> (0=CODE 39 pieno; 1=EAN8/13 pieno; 2=CODE 39 standard; 3=EAN8/13standard; 4=CODE128 pieno; 5=CODE 128 compresso) con stampa del codice articolo in base <prncod> (1=si,0=no) e l'unità di misura <um> (0 =millimetri, 1 =caratteri) da utilizzare nei parametri numerici <x>/<y> (distanza dal bordo sinistro/superiore del foglio) e nei parametri num. <cx>/<cy> (larghezza/altezza del barcode, con proporzione ottimale di 3 a 2).

Esempi.

```
LPRINTBARCODE arcod(),1,0,0,15,80,30,20
LPRINTBARCODE aralt(),0,1,0,20,20+40*COD,30,20 (<y> variabile in base a....)
LPRINTBARCODE CHR$(134)+"104512XA"+CHR$(134)+"17050101",4,1,0,0,0,50,40 (cod. EAN/UCC
128 con carattere FNC1 prima di ogni AI, nell'esempio il codice "4512XA" preceduto da AI"10" e
data "050101" preceduta da AI "17")
```

## LPRINTIMG

### LPRINTIMG <nomefile$>,<um>,<x>,<y>,<cx>,<cy>

In Sprix/Collage stampa, su stampanti configurate di tipo W, F, E e PDF, la immagine (formati ammessi bmp, wmf, jpg, png) dal nome <nomefile$> con unitadi misura <um> (0 =millimetri, 1 =caratteri) da utilizzare nei parametri numerici <x>/<y> (distanza dal bordo sinistro/superiore del foglio) e nei parametri num. <cx>/<cy> (larghezza/altezza dell'immagine, se entrambi = 0 significa una grandezza reale).

Esempio:

```
IMAGE$="c:\temp\immagine.bmp": LPRINTIMG IMAGE$,0,20,20,0,0
```

## LPRINTTXT

### LPRINTTXT <str$>,<font$>,<dimfont>,<attrifont$>,<um>,<x>,<y>,[opz$]

In Sprix/Collage stampa su stampanti configurate di tipo W, F e PDF la stringa <str$> con un font dal nome <font$> (disponibile e identico a quello nella cartella Windows), dimensione in numero di punti <dimfont>, gli attributi<attrifont$> combinabili ("G" = Grassetto, "I" = Corsivo, "S" = Sottolineato"" = Normale), l'unità di misura <um> (0=millimetri, 1=caratteri) da utilizzare nei parametri numerici <x>/<y> (distanza dal bordo sinistro/superiore del foglio). La stringa facoltativa [opz$] con i seguenti parametri combinabili: ALLINEAMENTO del testo (AS=sinistra(default), AC=centrato, AD=destra);STRETCH del testo (sia in altezza (SY) che larghezza (SX), entrambi seguiti dalla percentuale di stretch da 000 a 999 che se assente assume default 100,ad esempio, per raddoppiare l'altezza SY200 o dimezzare la larghezza SX050);il COLORE RGB del testo (con parametro C seguito da un numero di 3 cifre tra000 e 255, es. bianco = C255255255, nero = C000000000 -è default-).

Esempio

```
LPRINTTXT "Prova di stampa","Courier New",10,"GIS",0,15,15,"AC C255000000"
```

Non effettuabile in Linux spool, avvalersi di LPRINT e comandi della stampante (v. manuale stampante); in alternativa verificare le possibilità con gli stili di stampa (es. INIZCO$(),...) e LPRINT CHR$(...)+CHR$(...)+..., con file ".dfs" per ciascuna stampante.

## LUNGMOD

### LUNGMOD$(<numrighe>)

In Sprix/Collage ritorna una stringa di caratteri da inviare alla stampante per impostare la lunghezza del modulo di stampa con un numero di righe pari a <numrighe>. La lunghezza del modulo di carta è in numero di righe, in 6' di pollice. Es. LPRINT LUNGMOD$(66); (modulo impostato lungo 66 righe)

## MAXCOLLAGE

### MAXCOLLAGE()

In Collage ritorna la quantità di Collage caricati in memoria nel momento.

# MEXALDB

### MEXALDB <p1$>,<p2$>[,d$]

In Sprix esporta i dati sul database Mexaldb in chiaro (se abilitato) relativi agli archivi Generali <p1$> (si="S" o no="N") e Aziendali <p2$> dell'anno corrente "C" o di tutti gli anni "T" o nessuno "N". Se eseguita dall'aziendaSTD' avviene per tutte le aziende. Il parametro [,d$] è per esportare per differenza e quindi solo i variati ("SV=S" o "SV=N") e riguarda solo i documenti di magazzino e primanota con attivo l'aggiornamento replica dati in chiaro "M" manuale.

Esempio MEXALDB "S","T","SV=S"

MEXALDB valorizza il codice di esito dell'operazione _ERRMEXALDB ed il relativo messaggio _ERRMEXALDB$:

| | |
|---|---|
| 1 | terminale non bloccabile |
| 2 | da "STD" alcune aziende non esportate |
| 3 | azienda non esportata |
| 4 | parametri errati |

Nell'esecuzione via MXRSBATCH eventuali errori syserr a video sono indicati nel file degli errori stabilito dall'utente.

# MID

### MID$(<str$>,<posiniz>,<numcar>)

In Sprix/Collage estrae i primi caratteri <numcar> dalla stringa <str$> partendo dalla posizione iniziale <posiniz> dalla sinistra di <str$>; <posiniz>deve essere compreso fra 1 e la dimensione massima di <str$>; <numcar> non può essere 0 e se > della lunghezza <str$> la parte eccedente è ignorata.

Esempio:
A$="sprixEcollage": B$=MID$(A$,4,5) risultato B$="ixEco"
C$="sprixEcollage": D=4: E=LEN(C$)-D+1: F$=MID$(C$,D,E) ris. F$="ixEcollage"

# MKA

### MKA$(<str$>,<numbyte>)

In Sprix/Collage aggiunge tanti caratteri ASCII=0 in fondo a stringa <str$> fino a raggiungere la lunghezza totale di <numbyte>. Se <str$> è più lungadi <numbyte> viene troncata.

Esempio
A$="Sprix": B$=MKA$(A$,7) ris. B$="Sprix "

# MKN

### MKN$(<num>,<numbyte>,<numdec>)

In Sprix/Collage converte <num> (deve essere compreso tra 2^numbyte*8bit/2 e2^numbyte*8bit/2-1) in stringa usando la codifica binaria. Impacca <num> in meno byte rispetto a STR$; la lunghezza della stringa convertita <numbyte> deve essere compresa tra 1 e 5 e determina il limite massimo del valore da convertire (v. tabella) con numero di decimali <numdec> compreso tra 0 e 6 e per ogni decimale il limite va diviso per dieci; la stringa non è interpretabile direttamente e per riconvertirla in <num> occorre utilizzare CVN.

Esempio:
C$=MKN$(C,2,0): converte C (tra -32.768 e 32.767) in stringa 2 byte e 0 dec.

```
+----------------------------------------------------+
| TABELLA sviluppo cifre binarie per Num. Bytes      |
+----------------------------------------------------+
| Num. Bytes |         da         |         a        |
|------------|-------------------|-------------------|
|     1      |               -128 |               127 |
|     2      |            -32.768 |            32.767 |
|     3      |         -8.388.608 |         8.388.607 |
|     4      |     -2.147.483.648 |     2.147.483.647 |
|     5      | -549.755.813.888 |   549.755.813.887 |
+----------------------------------------------------+
```

# MYDBDEL

### MYDBDEL Archivio$

| | |
|---|---|
| Archivio$ | Archivio su cui determinare il numero delle chiavi<br>(sintassi: *CodiceProduttore_CodiceTabella/Anagrafica*) |

Esegue la cancellazione di un'anagrafica con ID uguale a _MYDBID

Viene ritornata la variabile _ERRMYDB che specifica il risultato dell'operazione.

## MYDBGET

### MYDBGET Archivio$, NumeroKey, Numero ID

| | |
|---|---|
| Archivio$ | Archivio su cui determinare il numero delle chiavi<br>(sintassi: *CodiceProduttore_CodiceTabella/Anagrafica*) |
| NumeroKey | 0 Esegue la lettura per ID<br>>= 1 Esegue la lettura ricercando la chiave impostata nell'array _MYDBK$()<br>= 99 Esegue la ricerca sull'estensione se dichiarata |
| Numero ID | se NumeroKey = 0, ID = id da leggere.<br>Se NumeroKey <> 0 legge il record dell'ID impostato |

Legge i dati dell'anagrafica dell'archivio impostato, per il valore della chiave o ID impostato. Ritorna i dati dell'anagrafica ricercata, o della successiva, se non trovata. Imposta le variabili _MYDBELE, _MYDB$(), _MYDBANN$, _MYDBETIC$(), _MYDBID e _MYDBNR.

Viene ritornata la variabile _ERRMYDB che specifica il risultato dell'operazione.

```
Esempio:
ARCHIVIO$ = "1900074000_interven"
'** Lettura di un'anagrafica dell'archivio
' valorizza il vettore _MYDB$()
' valorizza _MYDBID
' valorizza _MYDBANN$
MYDBGET "1900074000_interven", 0, 7
IF _ERRMYDB$ <> "" THEN VIMSG 5,_ERRMYDB$
Legge per l'archivio dichiarato il record con ID = 7
 _MYDBK$(1) = "501.00001"
MYDBGET "1900074000_interven", 99, 0
IF _ERRMYDB$ <> "" THEN VIMSG 5,_ERRMYDB$
Legge per l'archivio dichiarato il primo record dove la chiave 1 è = 501.00001
```

## MYDBGETATT

Lettura dell'anagrafica corrente

Esegue in automatico una **AZZVARSYS 35**

Imposta le variabili _MYDBELE, _MYDB$(), _MYDBANN$, _MYDBETIC$(), _MYDBID e _MYDBARC$

## MYDBINPUT

### MYDBINPUT Archivio$, NumeroID, [ Parametri$ ]

| | |
|---|---|
| Archivio$ | Archivio su cui determinare il numero delle chiavi<br>(sintassi: *CodiceProduttore_CodiceTabella/Anagrafica*) |
| Numero ID | = 0 Apre la finestra dell'archivo vuota, tutti i campi vuoti, o con i valori di default,<br>>=1 apre la finestra in Revisione dell'anagrafica con ID immesso,<br><0 apre la finestra dell'archivio mostrando a video i dati presenti nella struttura _MYDB$() |
| Parametri$ | "NOPUT" = Non esegue la scrittura e/o Cancellazione del record all'uscita della finestra |

Input a finestra di un'anagrafica. All'F10 viene eseguita una MYDBPUT.

```
Esempio:
MYDBINPUT ARCHIVIO$, 13, "NOPUT" '
Apre a video il record 13 ma non esegue la scrittura del record
MYDBINPUT "1900074000_interven", 13
Apre a video il record 13 ed esegue la registrazione dello stesso
```

## MYDBLOF

### MYDBLOF (Archivio$, " ")

| | |
|---|---|
| Archivio$ | Archivio su cui determinare il numero delle chiavi<br>(sintassi: *CodiceProduttore_CodiceTabella/Anagrafica*) |

Ritorna il numero delle chiavi inserite nell'archivio specificato.

Esempio:
NumeroChiave = MYDBLOF ("1900074000_interven", " ")
Ritorna il numero di chiavi nell'archivio indicato.

*ATTENZIONE: Tale funzione non può essere utilizzata all'interno di un ciclo di lettura del file (MYDBGET e MYDBNEXT/MYDBPREV) eseguire prima la MYDBLOF e poi le funzioni di lettura del file.*

# MYDBNEXT

### MYDBNEXT Archivio$, NumeroKey

| | |
|---|---|
| Archivio$ | Archivio su cui determinare il numero delle chiavi<br>(sintassi: *CodiceProduttore_CodiceTabella/Anagrafica*) |
| NumeroKey | 0 Esegue la lettura per ID<br>>= 1 Esegue la lettura ricercando la chiave impostata nell'array _MYDBK$()<br>= 99 Esegue la lettura per estensione |

Si posiziona sull'anagrafica successiva all'ultima anagrafica letta o scritta

Viene ritornata la variabile _ERRMYDB che specifica il risultato dell'operazione.

# MYDBPREV

### MYDBPREV Archivio$, NumeroKey

| | |
|---|---|
| Archivio$ | Archivio su cui determinare il numero delle chiavi<br>(sintassi: *CodiceProduttore_CodiceTabella/Anagrafica*) |
| NumeroKey | 0 Esegue la lettura per ID<br>>= 1 Esegue la lettura ricercando la chiave impostata nell'array _MYDBK$()<br>= 99 Esegue la lettura per estensione |

Si posiziona sull'anagrafica precedente all'ultima anagrafica letta o scritta

Viene ritornata la variabile _ERRMYDB che specifica il risultato dell'operazione.

# MYDBPUT

### MYDBPUT Archivio$

| | |
|---|---|
| Archivio$ | Archivio su cui determinare il numero delle chiavi<br>(sintassi: *CodiceProduttore_CodiceTabella/Anagrafica*) |

Esegue la scrittura/variazione di un'anagrafica, prende i dati dalle variabili _MYDB$(), _MYDBANN$ e _MYDBID. Imposta le variabili _MYDBID e _MYDBNR. Se _MYDBID == 0 esegue un inserimento.

Vengono ritornate le variabili _MYDBID, _MYDBNR e _ERRMYDB che specifica il risultato dell'operazione.

Esempio:
Letto un record con l'istruzione MYDBGET
' sul record specificato in _MYDBID
_MYDB$(10) = "Modificata da sprix"
MYDBPUT ARCHIVIO$

# MYDBPUTATT

### MYDBPUTATT Parametri$

| | |
|---|---|
| Parametri$ | "GOTO=CodiceCampo". Si posiziona sul campo, o se non possibile sul successivo disponibile, con il codice passato<br>"NOCTR". Non esegue i controlli di gestione sui vari campi |

Impostazione dell'anagrafica corrente. Legge il contenuto delle variabili _MYDB$(), _MYDBANN$ e sovrascrive (solo in memoria) l'anagrafica corrente. Poiché collage è richiamato da un campo di input, i nuovi dati saranno visibili all'uscita del campo corrente.

*N.B.: Il comando MYDBPUTATT non esegue l'aggiornamento del campo in cui il cursore è posizionato, per tale valorizzazione utilizzare la _SINP$.*

```
Esempio:
ON_OUT_YMYD001:
'Leggo le info dalla videata corrente e valorizzo l'array _MYDB$()
MYDBGETATT
_MYDB$(3) = "2" ' Tecnico
_MYDB$(4) = "1" ' Tipo intervento
_MYDB$(5) = "1" ' Stato intervento
_MYDB$(6) = sxdat ' Data
_MYDB$(7) = "5" ' Durata
_MYDB$(8) = "250,50" ' Prezzo
_MYDB$(9) = "N" ' Pagato
_MYDB$(10) = "Creato da Collage"' Note
' Imposto nella videata i valori impostati nell'array
' Mi posiziono sul campo Cliente
MYDBPUTATT "GOTO=005"
IF _ERRMYDB$ <> "" THEN VIMSG 5,_ERRMYDB$
END
```

# MYDBRIORG

### MYDBRIORG Archivio$, [Parametri$]

| | |
|---|---|
| Archivio$ | Archivio su cui fare la lettura (“*produttore_archivio*”) |
| Parametri$ | “SOLOKEY” Ricrea solo le chiavi, non controlla i dati<br>“SILENT” Non visualizza la finestra di avanzamento della riorganizzazione<br>“STAMP=*numero stampante*” Imposta la stampante sulla quale mandare l’esito della riorganizzazione. Se la stampante indicata non è disponibile, viene richiesta. |

Riorganizzazione di un archivio MyDB. Se lanciato senza parametri viene riorganizzato l’archivio passato (compreso il controllo dei dati) visualizzando la finestra di avanzamento.

**Poiché la riorganizzazione prevede la creazione delle chiavi, nessun terminale deve aver aperto tale archivio. Non viene controllato, quindi nel caso il programma chiude il termianale come fa con la OPEN** "K*".

Può tornare gli errori:

102 Archivio MyDB non definito 'Prod_Cod'

110 Archivio MyDB non trovato 'Prod_Cod'

# MYDBTAB

### MYDBTAB Archivio$

| | |
|---|---|
| Archivio$ | Archivio su cui determinare il numero delle chiavi<br>(sintassi: *CodiceProduttore_CodiceTabella/Anagrafica*) |

Imposta la tabella per il ciclo tabellare dell’archivio specificato. L’istruzione va dichiarata prima dell’esecuzione del ciclo tabellare, al di fuori del ciclo stesso.

```
Esempio
MYDBTAB “PRODUTTORE_ARCHIVIO”   10 caratteri del produttore + _ + 8 caratteri del nome
archivio
“ciclo tabellare yd”
```

# MYDBVIEW

### MYDBVIEW Archivio$, NumeroKey, NumeroID, [Parametri$]

| | |
|---|---|
| Archivio$ | Archivio su cui determinare il numero delle chiavi<br>(sintassi: *CodiceProduttore_CodiceTabella/Anagrafica*) |
| NumeroKey | = 0 Esegue la lettura per ID<br>>= 1 Esegue la lettura ricercando la chiave impostata nell’array _MYDBK$()<br>= 99 Esegue la lettura per estensione |
| Numero ID | < NumeroKey> uguale a 0 si posiziona sull’ID immesso,<br>< NumeroKey> diverso da 0 si posiziona sulla chiave immessa in _MYDBK$() ed a parità di chiavi |

| | |
|---|---|
| | eventualmente sul <Numero ID> immesso |
| Parametri$ | "INSER". Abilita l'inserimento di una nuova anagrafica<br>"FILTRO=A". Nessun filtro e visualizza anche gli annullati<br>"FILTRO=1". Usa come filtro i campi specificati nell'array _MYDBF$() (e NON considera gli annullati)<br>"FILTRO=2". Usa come filtro i campi specificati nell'array _MYDBF$() (e considera gli annullati)<br>"FILTRO=3,Numero". Usa come filtro il numero di Selezione salvato precedentemente nella gestione dell'anagrafica (e NON considera gli annullati)<br>"FILTRO=4,Numero". Usa come filtro il numero di Selezione salvato precedentemente nella gestione dell'anagrafica (e considera gli annullati)<br>Nel caso di filtro 3 e 4 se il filtro non viene trovato, **non** viene ritornato nessun errore, e viene letto tutto l'archivio. |

Apre la finestra di ricerca di un'archivio.

Se NumeroKey e NumeroID sono uguali a zero viene visualizzato tutto l'archivio.

Se si preme invio su una riga vengono automaticamente impostate le variabili come se si fosse eseguita una MYDBGET

```
Esempio:
'** Visualizzazione elenco dei record
MYDBVIEW "1900074000_interven", 0, 0
La finestra mostra tutti I record elencati per ID
 _MYDBK$(1) = "501.00002"
MYDBVIEW "1900074000_interven", 1, 0
La finestra mostra tutti I record elencati a partire dal codice 501.00002
  _MYDBF$(1)="501.00002"
  _MYDBF$(2)="Domenica"
  MYDBVIEW "1901536000_INTERVEN", 0, 0, "FILTRO=1"
La finestra mostra tutti i record filtrati per il primo elemento della archivio = a 501.00002 e
secondo elemento dell'archivio =a "Domenica"
```

*Nota bene:*

*Nel caso in cui l'elemento dell'array _MYDBF$() sia un campo dell'archivio stesso, si imposta il valore del campo.*

*Se l'elemento dell'array _MYDBF$(X) è una relazione ad un altro archivio o tabella MYDB, il valore da impostare nell'elemento _MYDBF$(X) è il _MYDBID dell'archivio relazionato.*

# MODBFIELD

### MODBFIELD <opz$>,<FROM row TO...>,<FROM col TO... AS varbuf [tipoimp]>, ...

In Sprix definisce per la pagina, la variabile e l'area del file da cui assegnare i dati alla variabile. Èsimile a FIELD con dimensione anche verticale<FROM rowstart TO rowend>,<FROM colstart TO colend AS varbuf [tipoimp]>,..(riga d'inizio e fine campo/i colonna d'inizio e fine campo e variabile associata). Le opzioni <opz$> del campo sono separate da punto e virgola ";" (obbl = se assegnato a 1 non fa attivare il flag _MODSCRIVI se il campo è vuoto; break = se assegnato a 1 imposta l'attivazione del flag _MODSCRIVI non più sui fine pagina ma sulla rottura del valore letto). Utilizzabile se è invocata MODBODY.

# MODBGET

In Sprix legge i dati dal file definito in MODOPEN e li assegna alle variabili definite con le MODFIELD.

# MODBODY

### MODBODY <opzioni$>, <FROM rowstart TO rowend>,<rows>

In Sprix definisce la gestione di un eventuale corpo pagina, con <opzioni$> del corpo pagina (parametro non ancora attivo) e con riga di inizio e fine corpo nell'ambito della pagina <FROM rowstart TO rowend> e righe fisiche della riga logica del corpo pagina <rows>.

# MODCLOSE

In Sprix chiude il file testo del modulo e tutta la gestione modulo.

# MODFIELD

### MODFIELD <opz$>,<FROM row TO...>,<FROM col TO... AS varbuf [tipoimp]>, ...

In Sprix definisce per la pagina, la variabile e l'area del file da cui assegnare i dati alla variabile. Èsimile a FIELD con dimensione anche verticale<FROM rowstart TO rowend>,<FROM colstart TO colend AS varbuf [tipoimp]>,..(riga d'inizio e fine campo/i colonna d'inizio e fine campo e variabile associata). Le opzioni <opz$> del campo sono separate da punto e virgola ";" (obbl = se assegnato a 1 non fa attivare il flag _MODSCRIVI se il campo è vuoto; break = se assegnato a 1 imposta l'attivazione del flag _MODSCRIVI non più sui fine pagina ma sulla rottura del valore letto).

# MODOPEN

### MODOPEN <tipo$>,<nomefile$>,<archivi$>,<opzioni$>,<righe>,<colonne>

In Sprix apre il file testo dal nome <nomefile$> relativo a modulo di stampa da importare di un certo <tipo$> (se = "S" è semplice con righe sequenziali senza pagine; se = "C" è complesso con pagine ed eventuale corpo pagina con la necessaria indicazione del numero di <righe> pagina) e con certo numero di <colonne>, al fine di importare il tipo di <archivi$> da gestire (archivi disponibili: "sc" = saldi contabili), in base alle seguenti <opzioni$> del modulo separate da punto e virgola ";" (es.: "des=Import Saldi;decsep=.")

| | |
|---|---|
| des | eventuale descrizione del modulo di import; |
| decsep | carattere separazione decimale , o . (il default è virgola); |
| cls | codice classificazione (es. "SC" Saldi conto); |
| newline | codici numerici per i separatori di riga (max 2), se non definito si utilizza lo standard ASCII 13,10 (CHR(13), CHR(10)); |

# MODPUT

### MODPUT <archivi$>

In Sprix scrive le variabili predefinite lette con MODGET negli <archivi$> definiti in MODOPEN e indicati con sigle in minuscolo separate da virgola (es. "sc" = saldi conto).

In caso di "sc", è attiva la variabile _PNNDO che, se impostata, scrive il documento assegnandogli il numero, creando le relative scadenze.

# MODSFIELD

### MODSFIELD <opz$>,<varbufori [tipoimp]>,<nomefile$>,<obbligo>,<varbufdesori$>

In Sprix permette di associare ad un campo una tabella di sostituzioni (vedere menu Servizi\Personalizzazioni\Importazione Moduli\Tipi Sostituzione), in base ad opzioni <opz$> (parametro non ancora attivo); la variabile per cui gestire le sostituzioni <varbufori [tipoimp]> deve esistere nelle MODFIELD /

MODBFIELD; <nomefile$> è il nome logico del file da gestire (conti predefiniti e generico); in <obbligo> se = "S" il flag _MODSCRIVI non viene attivato se la variabile NON ha un valore associato <varbufdesori$>, cioè l'eventuale variabile di descrizione di <varbufori> da presentare nella gestione delle assegnazioni sostituzioni (opzionale).

# MODVIMSG

In Sprix visualizza gli eventuali messaggi che sono contenuti nella variabile _MSGMOD$() alla fine di un'elaborazione modulo. Utilizzabile dopo MODOPEN

# NEXTKEY

### NEXTKEY <numfile>,<numkey>

In Sprix/Collage ritorna nelle variabili associate al buffer del file a indici aperto con numero <numfile>, il record SUCCESSIVO all'ultimo record letto(quello ritornato da GETKEY, NEXTKEY o PREVKEY) seguendo la chiave <numkey> (da 1 a 4, cioè una fra quella primaria e le secondarie). Se il record successivo non esiste tutte le variabili del buffer sono impostate con ASCII 0.Valorizza _NRDATI e _ERRIND.

# NOMECOLLAGE

### NOMECOLLAGE$(<num>)

In Collage ritorna il nome del file Collage con numero indice d'ordine <num>del Collage caricato in memoria (fra 1 e MAXCOLLAGE());

# ON_CTRLMYDB

### ON_CTRLMYDB:

L’evento è richiamato all’uscita di un’anagrafica. L’evento è in alternativa all’evento ON_CTRLFILTROMYDB.

# ON_CTRLFILTROMYDB

### ON_CTRLFILTROMYDB:

L’evento viene richiamato all’uscita del filtro di un’anagrafica.
Eseguito dopo i controlli del programma, ma prima dell ON_ENDMYDB.
Sono attivi MYDBGETATT e MYDBPUTATT e la varaibile _TF che contiente il codice del tasto di uscita.
Se si modifica _TF o si esegue la MYDBPUTATT, il programma reintra in anagrafica per visualizzare le modifiche e rieseguire i controlli sull’anagrafica.
Se si imposta _TF con un tasto valido, questo viene rielaborato.
L’evento è in alternativa all’evento ON_CTRLMYDB.

# ON_END

### ON_END:

al termine del modulo

# ON_ENDFILTROMYDB

### ON_ENDFILTROMYDB:

L’evento viene richiamato all’uscita del filtro di un’anagrafica.

# ON_LISTADOCMM

### ON_LISTADOCMM:

Questo evento si scatena sulla lista dei documenti in emissione revisione documenti. Tale evento viene attivato solo se i parametri “Compatibilità collage” e “Accesso emis./revis. Doc a lista ” sono rispettivamente a No e Si.

Il vettore _LISTADOCMM$(), contiene i 999 caratteri presenti a video. Tali informazioni, sono suddivise nei 4 slot dell’array (250+250+250+249 = 999).

# ON_SMSGARTMM

### ON_SMSGARTMM:

questo evento si scatena prima dell’apertura delle finestre di dettaglio articoli in emissione revisione documenti. Tale evento permette di gestire al meglio l’apertura e/o chiusura delle finestre rispettando la sequenza delle stesse.

# ON_START

### ON_START:

primo evento all'avvio del modulo che non sia 'AUTO o richiamato con STARTCOLLAGE dato che è processato all'effettivo caricamento in memoria salvo gestione nel codice;

# ON_STARTFILTROMYDB

### ON_STARTFILTROMYDB:

L’evento viene richiamato all’ingresso del filtro di un’anagrafica.

# ON_STARTMYDBRIC

### ON_STARTMYDBRIC:

L’evento viene richiamato all’ingresso di una ricerca di un archivio.

# ON_STP_DOC

### ON_STP_DOC:

Per le stampe personalizzate è attiva una sola label "ON_STP_DOC:".

Le uniche stampe personalizzate su cui è attivo il collage sono:
"Movimenti di Magazzino"
"Documenti di Magazzino"

Attivata la label nel sorgente collage la lettura delle informazioni per le righe elaborate avviene tramite l'esecuzione del comando "GETATTMM".

Le variabili da valorizzare sono:

_STPGXA$(1-20) : _STPGXN(1-20)

Variabili generali, valorizzabili e modificabili sempre. Definiscono un valore che sarà riportato alla fine dell'elaborazione nella relativa variabile di stampa

_STPTXA$(1-20) : _STPTXN(1-20)

Variabili di documento, attive per ogni riga elaborata dalla stampa.

_STPRXA$(X,1-10) : _STPRXN(X,1-10)

_STPRWS$(X,1-20) : _STPRWF(X,1-20)

Variabili bidimensionali (X = riga documento, elemento vettore). Attive per le righe del documento.

L'utilizzo delle variabili è identico a quello delle variabili collage _MMSWA$(_NMM,1-10) e _MMSWN(_NMM,1-10), a cui si rimanda per ulteriori dettagli.

Esempio:

```
ON_STP_DOC:
 _STPTNO = 0
 GETATTMM
 IF _MMSIG$ = "FT" THEN _STPTXA$(1) = "Valore personalizzato"
 FOR I = 1 TO _NMM
  IF _MMSIG$ = "FT" THEN _STPRXA$(I,1) = "Valore personalizzato"
  NEXT
  IF _MMSIG$ = "BD" THEN _STPTNO = 1
END
```

# OPEN

### OPEN <tipo$>,<numfile>,<nomefile$>[,nomefilek$;pos1;lun1;pos2;lun2;...],<lr>

In Sprix/Collage apre un file sequenziale (se <tipo$> = "R") o ad indici (se<tipo$> = "K") con riorganizzazione del file chiavi prima dell'apertura (se <tipo$> = "K*", consigliato se monoutenza altrimenti sarebbe meglio che fosse un'apposita funzione a parte).

Il file è creato con numero <numfile> da 1 fino a 15 contemporanei (massimo 15 se <tipo$>="R" e 5 se <tipo$>="K"/"K*"fino a un totale di 15 compresi i file chiavi) e ha nome <nomefile$>.

I restanti parametri servono quando <tipo$>="K"/"K*" per dichiarare e strutturare da 1 fino a 4 file di chiavi dal nome <nomefilek$> (numerabili progressivamente da sinistra a destra dal numero 1 obbligatorio e con la chiave principale univoca, fino ad altri 2,3,4 facoltativi e con chiavi secondarie), ciascuno con posizione e lunghezza <posN;lunN> della chiave all'interno del file<nomefile$> (con N da 1 fino a 4 posizioni per ciascun <nomefilek$> e la cui sommatoria dei vari 'lun' non può su perare 64 bytes - massima lunghezza di ciascuna chiave -).

Il tutto per lunghezza record <lr> del file <nomefile$> da 1 a 2048 bytes. Consentiti fino a 16.000.000 di records.

La lunghezza di un campo del record è 255 byte (equivalentemente alla variabile).

Terminate le operazioni della elaborazione del file successive all'apertura, CLOSE <numfile> in Sprix/Collage chiude il file numero <numfile> per renderlo disponibile. Il termine del programma chiude il file.

Esempi Sprix:

gestione di file ad accesso diretto e ad indici

```
VIMSG 1, "IMPORTO FILE HTML DA INTERNET"
GETINTERNET "[porta:80]www.passepartout.net","c:\temp\Pass.html"
VIMSG 1, "TEST SU DIRECTORY E LEGGO FILE"
ACCFILEUTE=1 : IF _ERRFILE$<>"" THEN VIMSG 4, _ERRFILE$ : END
LEGGIDIR "c:\temp","" : FOR F=1 TO _NUMDIRFILE
 WA_FILE$=WA_FILE$+"|"+_DIRFILE$(F) : NEXT : VIMSG 1, "<FILE:|"+WA_FILE$

VIMSG 1, "SCRIVO UN FILE SEQUENZIALE FORMATTATO"
F$="c:\temp\file.txt": CRLF$=CHR$(13)+CHR$(10)
NF=AUTOFILE() : OPEN "R",NF,F$,12:FIELD NF,10 AS REC$,2 AS CRLF$
```

```
FOR R=1 TO 9 : REC$=STRING$(R,STR$(R,0,0)) : PUT NF,R : NEXT : CLOSE NF
VIMSG 1, "LEGGO IL FILE SEQUENZIALE FORMATTATO"
OPEN "R",NF,F$,12 : FIELD NF,10 AS REC$,2 AS CRLF$
NREC=LOF(NF)/12 : DIM REC$() : FOR R=1 TO NREC : GET NF,R : REC$(R)=REC$
TREC$=TREC$+"|"+CVA$(REC$(R),10):NEXT:VIMSG 1, "<Record Totali:|"+TREC$

VIMSG 1, "AGGIUNGO DEI RECORD FORMATTATI"
 REC$="record 10" : PUT NF,10 : REC$="record 11" : PUT NF,11 : CLOSE NF

VIMSG 1, "AGGIUNGO RECORD (SOVRASCRIVO E PERDO FORMATTAZIONE -CRLF$-)"
OPEN "R",NF,F$,12 : FIELD NF,10 AS REC$
REC$="SPRIX" : PUT NF,4 : REC$="COLLAGE" : PUT NF,7
REC$="  MDS" : PUT NF,12 : REC$="  SDK" : PUT NF,13
CLOSE NF

VIMSG 1, "LEGGO IL FILE NON FORMATTATO|(-record dimensione variabile-)"
OPEN "R",NF,F$,1:FIELD NF,1 AS CAR$:DIMFILE=LOF(NF):RECORD$="":R=0
FOR C=1 TO DIMFILE: GET NF,C
IF CAR$<>CHR$(13) AND CAR$<>CHR$(10) THEN GOTO CONCATENA
IF CAR$=CHR$(13) THEN GOTO PROSSIMOREC
IF CAR$=CHR$(10) THEN GOTO VERIFICA
CONCATENA: RECORD$=RECORD$+CAR$ : GOTO PROSSIMOREC
VERIFICA:R=R+1
IF INSTR(RECORD$,"333")<>0 THEN VIMSG 1,"333|è al record "+STR$(R,0,0)
RECORD$=""
PROSSIMOREC: NEXT : CLOSE NF

VIMSG 1, "CREO UNA COPIA DI BACKUP DEL FILE SEQUENZIALE E LO CANCELLO"
IF ESISFILE(F$) THEN COPYFILE F$,"c:\temp\file_R_back.txt" : KILL F$

VIMSG 1, "CREO FILE A 2 CHIAVI CON:| Articolo, Descrizione, Categoria"
F$="c:\temp\file.txt":F1K$="c:\temp\file1.k":F2K$="c:\temp\file2.k"
OPEN "K",NF,F$,F1K$;1;16,F2K$;41;10,52
FIELD NF,20 AS ART$ ALFA,20 AS DES$ ALFA
FIELD NF,10 AS CAT NUM0,2 AS CRLF$ ALFA
FOR I=1 TO 9: I$=STR$(I,0,0)
ART$="Articolo n."+I$: DES$="Descrizione "+I$: CAT=VAL(STRING$(I,I$))
INSKEY NF : IF _ERRIND=1 THEN VIMSG 4,"record già presente"
NEXT: CLOSE NF

VIMSG 4, "CERCO|Articoli n.10 e n.5|PER CHIAVE PRIMARIA E LI MODIFICO"
 OPEN "K",NF,F$,F1K$;1;16,F2K$;41;10,52
FIELD NF,20 AS ART$ ALFA,20 AS DES$ ALFA,10 AS CAT NUM0,2 AS CRLF$ ALFA
KEY$="Articolo n.10": GETKEY NF,1,KEY$
IF _ERRIND=1 THEN VIMSG 4,"articolo 10 cercato ma non trovato":
NEWKEY$="Articolo n.5": GETKEY NF,1,NEWKEY$
IF _ERRIND=0 THEN VIMSG 1,"articolo 5 trovato, gli cambio descrizione"
NEWDES$=RIGHT$(STRING$(5,"NUOVO"),20):DES$=NEWDES$
VARKEY NF, NEWKEY$: IF _ERRIND=0 THEN VIMSG 1, "Descrizione 5 variata"

VIMSG 1, "ORA LEGGO PROSSIMI RECORD E VARIO DESCRIZIONI"
 CICLO: NEXTKEY NF,1: IF _ERRIND=1 THEN GOTO FINEFILE
KEYART$=LEFT$(ART$+STRING$(20,CHR$(0)),20)
DES$=NEWDES$:VARKEY NF, KEYART$:IF _ERRIND=1 THEN VIMSG 4,"non variati"
IF _ERRIND=1 THEN VIMSG 4,"record non variato"
GOTO CICLO
FINEFILE: CLOSE NF: VIMSG 1, "file terminato"

VIMSG 4, "LEGGO A RITROSO PER CHIAVE NUMERICA SECONDARIA"
LOCATE 5,2:PRINT "Leggi dalla categoria 1,22,333 o 4444..: ";:
INPUT "N",CERCAT,5: CERCAT$=RIGHT$("          "+STR$(CERCAT,0,0),10)
```

```
OPEN "K",NF,F$,F1K$;1;16,F2K$;41;10,52
FIELD NF,20 AS ART$ ALFA,20 AS DES$ ALFA,10 AS CAT NUM0,2 AS CRLF$ ALFA

VIMSG 1, "chiavi totali: "+ STR$(LOFKEY(NF),0,0)
KEY$=RIGHT$(CERCAT$,10):GETKEY NF,2,KEY$
IF _ERRIND=0 THEN VIMSG 1,"Numero Record Dati: "+STR$(_NRDATI,0,0)
CICLOPREV: PREVKEY NF,2: IF _ERRIND=1 THEN GOTO INIZFILE

VIMSG 4, "Key Precedente  "+STR$(CAT,0,0)+": con "+DES$: GOTO CICLOPREV
INIZFILE: VIMSG 1, "lettura termine file effettuata": CLS
T1$="ADESSO CERCA PER CATEGORIA (KEY2)"
T2$="SE TROVA VARIA, ALTRIMENTI INSERISCE NUOVO RECORD"
 T3$="E CANCELLA IL RECORD CON KEY1 |'Articolo n.5       '"
VIMSG 1,T1$+"|"+T2$+"|"+T3$

VARIAZIONE:
LOCATE 5,2:PRINT "Cerca Categoria (5 numeri).: ";:
INPUT "N",CERCAT,5: CERCAT$=RIGHT$("         "+STR$(CERCAT,0,0),10)
LOCATE 6,2:PRINT "Nuova Categoria (5 numeri).: ";:
INPUT "N",NEWCAT,5

GETKEY NF,2,CERCAT$: OLDART$=ART$ 'memorizzo la key principale
  IF _ERRIND=1 THEN TROV$="N" ELSE TROV$="S"

GOSUB CTRL: IF CTRL$<>"OK" THEN GOTO VARIAZIONE
  IF TROV$="S" THEN VARKEY NF,OLDART$:CLOSE NF:GOTO CHECK
  IF TROV$="N" THEN INSKEY NF: DELKEY NF, "Articolo n.5":CLOSE NF
  '(VARKEY mantiene il format su editor rispetto a DELKEY+INSKEY)

CHECK: IF _ERRIND<>0 THEN VIMSG 4, "problemi in VAR/INS/DEL KEY"
END

GOTO VARIAZIONE:
 CTRL: VIMSG 1, "effettuo controlli/modifiche..."
 CAT=VAL("     ")+NEWCAT
 ART$="NEW ARTICOLO  ????? ":DES$="NEW DESCRIZIONE ??? "
 CTRL$="OK": RETURN
```

# OPENCOM

### OPENCOM <com$>,<baudrate>,<parity>,<size$>,<stopbit>,<timeout>,<flag$>

In Sprix/Collage apre la porta seriale stringa <com$> (una sola alla volta),con velocità <baudrate> (110,300,600,1200,2400,4800,9600,19200,38400), parità <parity> (0=nessuna, 1=dispari, 2=pari), stringa con numero di bit della parola <size$> ("5","6","7","8"), flag bit di stop <stopbit> (0 significa 1 stopbit; 2 significa 2 stopbit), numero di secondi di attesa <timeout> in base al sistema operativo (0 significa 10 secondi; se preceduti da '-' significa millesimi in Windows o decimi in Linux). Flag stringa facoltativa <flag$>per specificare il comportamento di lettura del buffer in caso di un timeout(se "0" il buffer non è valorizzato -default-; se "1" il buffer contiene i byte letti fino al momento del timeout). Default in caso di parametri non validi: OPENCOM <com$>,9600,2,"7",0,0. Valorizza _ERRCOM$.

# PIUDATA

### PIUDATA$(<data$>,<numgior>)

In Sprix/Collage aumenta/diminuisce la stringa <data$> in formato "AAAAMMGG" compreso fra 19600101 e 20481231, del numero di giorni <numgior> compreso  fra -32.507 e 32.507 (se >0 aumenta, se <0 diminuisce).

```
Esempio
D$="20090621"
B$=PIUDATA$(D$,-1): C$=STRDATA$(PIUDATA$(B$,10),2,"/") risultato C$=30/06/09
```

# PREVKEY

### PREVKEY <numfile>,<numkey>

In Sprix/Collage ritorna nelle variabili associate al buffer del file a indici aperto con numero <numfile>, il record PRECEDENTE all'ultimo record letto(quello ritornato da GETKEY, NEXTKEY o PREVKEY) seguendo la chiave <numkey> (da 1 a 4, cioè una fra quella primaria e le secondarie). Se il record successivo non esiste tutte le variabili del buffer sono impostate con ASCII 0.Valorizza _NRDATI e _ERRIND.

# PRINT

### PRINT <espr$>[;]

In Sprix/Collage visualizza l'espressione della stringa <espr$> nella finestra attiva alla posizione e colore attuale. Se assente [;], dopo la visualizzazione, il cursore salta alla prima colonna nella riga successiva della finestra attiva (e se la visualizzazione è sull'ultima riga della finestra avviene lo scrolling di una riga verso l'alto). Non deve eccedere i limiti della finestra attiva e non consente scelta del font.

Esempio:

```
A$="AA": P=12: N=12: M=-21: PRINT A$+"Bb";: PRINT N+M;: PRINT " "+A$:PRINT P
```

# PRINTD

### PRINTD <str$...:>

In interfaccia grafica Sprix/Collage, visualizza evidenzia la stringa <str$>in modalità 'descrizione del campò; elimina i puntini e due punti finali e non manda il cursore a capo (occorre LOCATE).

# PRINTF

### PRINTF <n>,<nrip>

In Sprix/Collage visualizza il carattere <n> di fincatura per <nrip> volte. <n> vale 1(orizzontale), 2(verticale), 3(angolo sup. sin.), 4(angolo super. des.), 5(angolo inf. sin.), 6(angolo inf. des.), 7(incrocio) 8(raccordo in alto), 9(raccordo in basso),10(raccordo a sin.), 11(raccordo a destra). Alternative STRING$ o PRINT.

# PRINTH

In Sprix/Collage forza in modo incondizionato la stampa delle righe di intestazione (righe 'H') sulla stampante aperta.

# PUT

### PUT <numfile>,<numrec>

In Sprix/Collage scrive il numero di record <numrec> nel file aperto di tipo "R" numero <numfile>, utilizzando le variabili del buffer associato: in lettura sono riempite con i relativi valori letti (oppure tutte impostate a "" o 0 se il <numrec> è < 1 o > al numero di record presenti nel file). In scrittura, se la lunghezza dei valori delle variabili è > alle dimensioni dei buffer viene scritta solo la parte non eccedente, se è < vengono scritti i caratteri ASCII 0 per la parte eccedente.

Se <numrec> è < 1 la PUT non è eseguita, se <numrec> è > al numero di record presenti nel file, vengono accodati al file tanti record con i valori ASCII 0 fino raggiungere il numero di record richiesti (allargamento totale massimo 32.767 caratteri).

# PUTABBIN

In Sprix scrive/aggiorna i dati dei Codici Abbinati relativi alle variabili di struttura archivi _AB? in base al codice articolo <codart$> a cui sono abbinati

La lettura rilascia _NAB cioè il numero di elementi dell'array dinamico della struttura _AB?

# PUTAGP

### PUTAGP [par1]

In Sprix scrive i dati delle righe della AGENDA DI LAVORO, relativi alle variabili di struttura archivi _AGP?, in base ai seguenti parametri identificativi obbligatori.

In PUTAGP, par1 indica l'elemento dell'array _AGP?() che corrisponde alla registrazione da effettuare;

# PUTALIAS

In Sprix scrive/aggiorna (PUT) i dati dei Codici Alias relativi alle variabili di struttura archivi _AL? sulla base al codice articolo <codart$> a cui sono associati (se "" =tutti gli articoli), al codice alias <codalias$> (se "" = tutti gli alias) e al numero progressivo per i codici alias doppi <nprog> (se -1 = tutti, altrimenti specificare il numero di chiave che si intende prelevare: 0,1,2,..).

La lettura rilascia la _NAL per conoscere il numero di elementi dell'array dinamico della struttura _AL?

# PUTAR

### PUTAR <nummag>

In Sprix scrive/aggiorna (PUT) i dati di Anagrafica Articoli relativi alle variabili di struttura archivi _ARCOD$ e gli eventuali progressivi del numero magazzino <nummag>. La PUTAR di articolo nuovo richiede i dati obbligatori della versione del gestionale in uso; la PUTAR di un articolo esistente lo aggiorna (valorizzare sempre tutte le variabili di interesse, altrimenti alcune potrebbero azzerarsi). I progressivi sono in sola lettura (eccetto l'inventario nel primo anno del articolo) e modificabili solo con documenti rettificativi.

# PUTAREA

In Sprix scrive/aggiorna i dati di Anagrafica Aree relativi alle variabili struttura archivi _AE?, in base al codice <Area>. La possibilità di cancellazione è attraverso il modulo Servizi di CdG.

# PUTASSOCDV

### PUTASSOCDV <tipo>,<keypass>,<iddoc>,[alleg]

In Sprix inserisce/aggiorna i dati delle associazioni fra il Documento Docuvision e l'Entità del Gestionale, relativi alle variabili di struttura archivi _AD?, in base all'ID del documento Docuvision <idDoc> (in DEL se <idDoc>=0 le elimina tutte) e il <tipo> di Entità del Gestionale, determinata univocamente dalla <keypass> di ricerca sulle tabelle d'associazione fra Gestionale-Docuvision. Le Entità del Gestionale appartenenti ad archivi annuali devono essere relative all'anno aperto.

Il parametro facoltativo [alleg] consente di variare la tipologia di allegato e il progressivo immagine attraverso la PUTASSOCDV. Il parametro ha la seguente sintassi "TPALL=x;PRGALL=y" dove "x" può assumere i seguenti valori null=Generico o I=Immagine o S=Scheda) o D=Descrizione e dove "y" può assumere valore numerico relativo al progressivo. Essendo parametro facoltativo,l'omissione di [alleg] equivale al default Generico "TPALL=;". Esempio di cambio di allegati Generici (associati ad "ART") in allegati immagine:

```
GETASSOCDV 1200,"ART",0
FOR X=1 TO _ADNUMASSOC
PUTASSOCDV 1200,"ART1",_ADDVDID(X),"TPALL=I;PRGALL="+STR$(X,0,0)
NEXT
```

GETASSOCDV valorizza _ADNUMASSOC che ritorna il num. globale d'associazioni.

La GET-PUT-DEL elaborano i dati _AD? in base ai seguenti parametri (eccetto per le classi <tipo>=1 Stampa generica, =800 Dichiarativi/Versamenti, =900 Stampe Fiscali; =1000 Agenda di lavoro ed eccetto i casi in cui il livello di azienda o il codice contratto non consentono l'accesso ad alcune tabelle,come es le operazioni di classe 'Parcella/Notulà senza gestione parcellazione, classi di Magazzino con azienda Contabile,...):

```
Piano dei Conti: <tipo>=400
  <keypass>="<codcon$>" (codice conto del piano dei conti)

Primanota: <tipo>=600
  <keypass> = "<dreg$>;<prinr>;<cau$>;<protr$>;<prots>;<protn>;<docn>;<ddoc$>"
                  <dreg$> (data di registrazione);
                  <prinr> (n. progressivo registrazione primanota);
                  <cau$>  (causale del documento);
                  <protr$>(registro di protocollazione)
                  <prots> (serie del protocollo)
                  <protn> (numero di protocollazione)
                  <docn>  (numero del documento)
                  <ddoc>  (data documento)

Anagrafica Articoli: <tipo>=1200
  <keypass> = "<codart$>" (codice articolo)

Movimenti di Magazzino: <tipo>=1400
  <keypass> = "<sig$>;<ser>;<num>;<clifor$>;<ddoc$>"
                  <sig$>    (sigla documento)
                  <ser>     (serie del documento)
                  <num>     (numero del documento)
                  <clifor$> (codice conto intestatario)
                  <ddoc$>   (data documento)

Ordine/Preventivi/Matrici: <tipo>=1500
  <keypass>="<sig$>;<ser>;<num>;<ddoc$>"
                  <sig$>  (sigla documento)
```

```
                    <ser>   (serie del documento)
                    <num>   (numero del documento)
                    <ddoc$> (data documento)

Bolla di Lavoro: <tipo>=1600
  <keypass> = "<numeroBL>;<numeroSottoBL>"
                    <numeroBL>      (numero di bolla di lavoro)
                    <numeroSottoBL> (numero di sottobolla di lavoro)
                    Inserire "-1" per avere tutte le sottobolle, inserire "0" (o non specificare) per intendere la BL principale. In
                    operazione DEL è richiesta anche la data della bolla.

Notule/Parcelle: <tipo>=1100
  <keypass>="<cau$>;<ser>;<num>;<clifor$>;<ddoc$>"
                    <cau$>    (causale del documento)
                    <ser>     (serie del documento)
                    <num>     (numero del documento)
                    <clifor$> (codice conto)
                    <ddoc>    (data documento)
```

## PUTBL

In Sprix inserisce/aggiorna (PUT) i dati di Bolla di Lavorazione relativi alle variabili _OB? (testata) e _OL? (righe)

Il Collage legge le _OB? e _OL? con la GETATTBL (senza parametri): azzera le variabili preesistenti e carica dati presenti a video fra evento START e END e fra PUT e DEL.

La generazione di testata senza dettaglio e senza errori, significa che le variabili di riga non sono state tutte impostate oppure che il numero della riga in produzione per il magazzino non è stato valorizzato.

Non si possono scrivere BL collegate a OC o sottobolla; per collegare vedere instruzione COLOCBL

La PUTBL non prevede nessun tipo di automatismo presente nella funzione standard della procedura

## PUTCDG

Scrive i dati dei movimenti di Contabilità Analitica relativi alle variabili _CDG?

La PUTCDG con codice commessa inesistente crea una nuova anagrafica commessa con codice e descrizione indicati; nel tipo 'R' avviene l'accodamento delle righe di analitica alle precedenti (l'aggiornamento richiede quindi una preventiva cancellazione).

## PUTCFR

Scrive i riferimenti dei Referenti presenti nell'anagrafica della rubrica Clienti Fornitori (tasto funzione F4) relativi alle variabili di struttura _CFR? in relazione ai valori del Cliente/Fornitore <cod$> in base al <tipo> (se 0 = tutti i referenti, da 1 a 9 = un referente specifico). la scrittura inserisce nuovi referenti o aggiorna i dati di quelli preesistenti, che restano inalterati se non diversamente assegnati dalle variabili Sprix (che quindi possono non essere valorizzate). Se non tutti i referenti sono valorizzati, _CFRPRG() non corrisponde con la posizione dei referenti all'interno della finestra.

Il collegamento fra referente registrato e array Sprix è stabilito da _CFRPRG().

Ad esempio:

| Array visibile a video | Array Sprix |
|---|---|
| 1 = Direzione | 1 = Direzione |
| 2,3,4,5 = “ ” | 2 = Amministrazione |
| 6 = Amministrazione | 3 = Assistenza |
| 7,8 = “ ” | |
| 9 = Assistenza | |

In Collage legge le variabili _CFR? con GETATTCFR (senza parametri); carica i dati presenti a video fra gli eventi ON_STARTCFR e ON_ENDCFR.

## PUTCOM

### PUTCOM <a$>,<n>

In Sprix/Collage scrive sulla porta seriale aperta, la quantità di <n> byte (da 1 a 255, con default 1 se < 1 o 255 se > 255) prelevati dal buffer dati <a$>. Valorizza _ERRCOM$.

## PUTCOMM

In Sprix scrive/aggiorna (PUT) i dati di Anagrafica Commessa relativi alle variabili struttura archivi _CO?, in base al codice <Commessa>.

# PUTCON

In Sprix inserisce i dati di Anagrafica Contatti relativi alle variabili _DC? in base al codice contatto <codice$>

# PUTDB

### PUTDB <numfase>

In Sprix scrive i dati della Distinta Base Primaria relativi alle variabili di struttura archivi _TP? in base al numero di fase <numfase> della DBP (1-99) e del codice articolo <codart$> con DBP.

# PUTDVDOC

In Sprix inserisce/aggiorna (PUT) i dati del Documento Docuvision relativi alle variabili struttura archivi _DV?, in base al numero Documento di Docuvision <numdoc> e numero revisione <numrev> (in lettura se -1=tutte le revisioni, 0=ultima revisione, >0=revisione specificata;in cancellazione se 0=cancella l'intero documento).

# PUTIL

In Sprix inserisce/aggiorna (PUT) i dati degli Impegni di Lavorazione relativi alle variabili di struttura archivi _IL?

Se in scrittura viene cambiata una _IL? della chiave dellaBL, viene inserita una nuova riga in Impegni di Lavorazione, identica a quella di partenza salvo per i valori cambiati; la scrittura non avviene se l'indice del vettore viene aumentato a più 1 per fare un nuovo inserimento; inserendo _ILNRB()=65534 o 65535 si può inserire un RES_SL o RES_DL.

# PUTIND

In Sprix inserisce/aggiorna i dati di Anagrafica Indirizzi di Spedizione relativi alle variabili di struttura archivi _DA? in base al codice indirizzo di spedizione <codice$> e all'eventuale [codice_conto$] cui è associato l'indirizzo ("" se non utilizzato).

# PUTLOT

In Sprix scrive i dati di Anagrafica Lotto relativi alle variabili struttura archivi _LOT? in base al numero ID lotto <idlot>. Nella scrittura il dato personalizzato si azzera se valorizzato null"".

# PUTMM

### PUTMM <scprn>

La scrittura necessita dei dati obbligatori richiesti dalla versione del gestionale in uso; il parametro numerico <scprn> consente di stabilire il tipo di registrazione:

| | |
|---|---|
| 0 | in magazzino; |
| 1 | in magazzino e primanota; |
| 2 | in magazzino con stampa; |
| 3 | in magazzino e primanota, con stampa; |
| 4 | stampa senza registrazioni. |
| 10 | uguale a 0 ma relativo a beni usati; |
| 11 | uguale a 1 ma relativo a beni usati; |
| 12 | uguale a 2 ma relativo a beni usati; |
| 13 | uguale a 3 ma relativo a beni usati; |

In Sprix scrive i dati del Documento di Magazzino relativi alle variabili struttura archivi _MM?

> *Notare: nella composizione del documento, la valorizzazione delle variabili di testata (quelle riferite alla _NTMM) devono essere valorizzate in modo uniforme e omogeneo, altrimenti non vengono stampate. (v. approfondimenti su operatività come da manuale utente).*

La primanota è registrata se prevista dal documento, che sia nuovo e diverso dalla FF.

Se il destinatario del documento gestisce la fattura elettronica o la fattura PA (Pubblica Amministrazione), il numero di serie _MMSER è prelevato in automatico.

In ristampa _MMRICSTP consente di attivare la richiesta di selezione stampante (se = 1), non attivare (se = 0, default) o stampare direttamente (se = -1); non è consentito indicare il num. di stampante (come da configurazione) e non è consentita in Shaker.

Consentite fino a 65.000 righe sul documento sospeso; il numero massimo di righe (R) deve rientrare nei limiti dell'array numerico e dipende dal tipo di installazione in uso ed è inversamente proporzionale alla quantità di informazioni aggiunte alla riga (videate personalizzate, imballi, variazioni di aliquota iva, taglie, modifiche alle descrizioni articoli,...). Nel movimento la quantità dipende dalla memoria del sistema. Nel caso, effettuare i test con quantità crescenti (1000, 2000,...) in varie condizioni di ulteriori dati. L'array (R) riferito alle righe del documento contiene inoltre i dati di testata dei (T) possibili documenti d'origine (max. 200) che possono comporre il documento.

La scrittura di un ordine prevede l'automatica registrazione della matrice se il parametro di magazzino "creazione automatica ordine matrice" = "S", altrimenti, se = "R" o la matrice esiste già viene aperta una finestra per la scelta (S/N): questa apertura si gestisce in Sprix con SETPARCREAZMATR<str$>e SETPARSOSTMATR<str$>, dove <str$> assume valore "N" o "S", per gestire il messaggio di Creazione o Sostituzione automatica a "Richiesta" in modo prioritario rispetto a quanto stabilito dal parametro.

Il Collage legge le _MM? con GETATTMM (senza parametri): azzera le variabili preesistenti e carica dati presenti a video fra evento START e END e fra PUTe DEL (lo START si attiva dopo l'uscita dal primo campo che divide l'emissione dalla revisione, mentre in emissione differita si attiva con trasformazione del documento e non con la ristampa).

## PUTPART

In Sprix scrive/aggiorna (PUT) i dati delle Particolarità Conti e Articoli relativi alle variabili struttura archivi _PART?,in base al tipo particolarità <tppart$>(P,S,V,A), al codice conto <codcon$> (o in alternativa al suo numero di categoria <catcon> listino, sconto, provvigione), al codice articolo <codart$> (o in alternativa al suo numero di categoria <catart> formazione prezzo, sconto, provvigione) e con data d'inizio validità <dtvalda$> della particolarità. La combinazione di quattro parametri individua la specifica particolarità e di questi, il secondo e il terzo sono combinabili in quattro modi: (<codcon$>,<codart$>) ;(<codcon$>,<catart>); (<catcon>,<codart$>); (<catcon>,<catart>). La PUTPART di particolarità nuova richiede i dati obbligatori della versione del gestionale in uso; se la particolarità esiste, viene aggiornata.

Collage legge anche con GETATTPART (senza parametri): azzera le variabili preesistenti e carica i dati presenti a video fra evento START e END e fra  PUT e DEL.

## PUTPC

Scrive  i dati del Piano dei Conti relativi alle variabili di struttura _PC? in base al codice conto <codcon$>.

La PUTPC del conto nuovo richiede i dati obbligatori della versione del gestionale in uso; la PUTPC di un conto già esistente lo aggiorna (valorizzare sempre tutte le variabili di interesse, altrimenti alcune potrebbero azzerarsi). Prestare anche attenzione alle informazioni relative al caricamento dei conti tramite la trasmissione CSV.

ATTENZIONE: in ambiente a file separati (tecnologia "BP"), per le aziende di gruppo, la PUTPC non è fruibile sui conti diversi da quelli di tipo C/F/B, perchè gli unici conti che questa processa sono i clienti, fornitori e banche (cioè quelli propri dell'azienda in questione). Tramite la variabile  _PCSOLOPICO$ = "S/N" (con default = "N") si possono aggiornare tutte le variabili tipiche del conto aziendale anche per i conti di raggruppamento.

I dati della rubrica (RUDT) rimangono invariati. Per i SOLI conti presenti nell'azienda le variabili che possono essere modificate quando la variabile _PCSOLOPICO$ è impostata = "S", sono contrassegnate con il simbolo (*a)   nella tabella di variabili _PC? elencate le relativo paragrafo.

## PUTPN

In Sprix scrive i dati della registrazione di Primanota relativi alle variabili di struttura archivi _PN?, in base alla data di registrazione <dreg$>, causale del documento <cau$>, registro di protocollo <protr$>, serie di protocollo <prots>, numero della protocollazione <protn>, numero del documento <docn>, data del documento <ddoc$>.

La PUTPN è solo per la nuova registrazione nella valuta di gestione azienda e con i dati obbligatori della versione del gestionale in uso.

La PUTPN non aggiorna il documento: occorre leggere la registrazione, cancellarla e reinserirla. La cancellazione mantiene i dati da reinserire, ma azzera l'eventuale collegamento al documento di magazzino (se ha generato la primanota), che serve per l'aggiornamento da magazzino a primanota (nel caso replicare la stessa operazione anche per il documento di magazzino, considerando che la cancellazione Sprix di magazzino non cancella la primanota) e con nuova assegnazione <prinr>.

La PUTPN aggiorna lo SCADENZARIO A DOCUMENTO dei Clienti/Fornitori presenti nella registrazione. L'importo da scadenzare POSITIVO (fattura) genera una o più scadenze in base al codice pagamento (se _PNPAG=0 è assunto quello di anagrafica Cli/For - tranne l'alternativo - e se è ancora zero è emessa un’unica scadenza con rimessa diretta 'M'). Non è attivo l'automatismo di pagamento in contanti: registrare un doc. PG. In caso di importo NEGATIVO (il pagamento) la _PNPCA$ determina il tipo di azione: se _PNPCA$ è valida allora la _PNPSD (per documenti sui Registri protocollo Iva "A","V","S","C"), la_PNPND e la _PNPDD$ del documento da pagare sono obbligatorie e vengono pagate tutte le rate fino a raggiungere l'importo in pagamento (se il documento non viene trovato o se l'importo da pagare è maggiore della somma delle rate da pagare viene emessa una scadenza negativa con causale rimessa diretta;se _PNPCA$ contiene un tipo di scadenza

valido ("C","M","X","Y","Z","R","B","A","T","V") oppure il carattere "*", vengono pagate tutte le rate con quel tipo di scadenza (o rate di qualsiasi tipo se _PNPCA$ è "*"), iniziando dalla più vecchia fino a raggiungere l'importo da pagare (l'eventuale residuo viene messo a pareggio); se _PNPCA$ è vuoto viene emessa una scadenza negativa con causale rimessa diretta. In tutti i casi in cui è previsto che vengano emesse una o più scadenze, se _PNPSD, _PNPND o _PNPDD$ del doc. sono a zero oppure non validi l'importo viene pareggiato. Lo SCADENZARIO A PARTITE (e relative _PN?) ha logica identica ed ogni documento ha la sua partita; la registrazione del pagamento può anche scegliere la partita da pagare e con lo stesso pagamento si possono pagare più documenti.

Collage legge le variabili _PN? con la GETATTPN (senza parametri): azzera le variabili preesistenti e carica i dati presenti a video fra l'evento START e END e fra la PUT (anche proveniente da registrazioni di magazzino) e la DEL.

# PUTPNX

La PUTPN è solo per la nuova registrazione nella valuta di gestione azienda e con i dati obbligatori della versione del gestionale in uso. Le contropartite AVERE sono < 0 e quelle DARE > 0. Conversioni Euro <-> Lire avvengono con PUTPNX: in valuta contabile Euro consente di esprimere importi in Lire e viceversa; non consente registrazione nella stessa valuta o in doppia valuta (ma solo in quella contabile).

# PUTSC

In Sprix scrive o cancella registrazioni di primanota al pari di quanto avviene per le istruzioni PUTPN / DELPN

# PUTSORT

## PUTSORT <str$>

In Sprix/Collage inserisce la stringa <str$> in una lista di record da ordinare. La _SORT_STR_CI stabilita all'inizio del programma stabilisce il tipo di ordinamento in caso di caratteri estesi (es. lettere accentate): è case sensitive se = 0 (default) o case insensitive se = 1.

La lunghezza in caratteri delle stringhe da ordinare, deve essere preventivamente definita con _LRECSORT (lunghezza max della <str$> da ordinare) e la velocità di ordinamento è inversamente proporzionale alla lunghezza; se è viene troncata, se è < sono aggiunti spazi bianchi.

La successiva lettura avviene con GETSORT <str$> che in Sprix/Collage legge la prossima stringa <str$> dalla lista di record ordinati da PUTSORT. Quando tutti record sono stati letti in <str$> è impostato il valore "". La chiamata della PUTSORT successiva alla GETSORT azzera tutti record precedentemente inseriti ordinati.

```
Esempio:
LRECSORT=8
PUTSORT"SPRIX 2": PUTSORT"SPRIX 6": PUTSORT"SPRIX 5": PUTSORT"SPRIX 1"
PUTSORT" ddd ": PUTSORT" bbb ": PUTSORT" aaa ": PUTSORT" ccc "
LEGGI:
GETSORT NOMEPR$ : PRINT NOMEPR$: VIMSG 1, NOMEPR$
  IF NOMEPR$="" THEN GOTO FINE ELSE GOTO LEGGI
  FINE: VIMSG 4, " fine ": END
```

# PUTTAST

## PUTTAST <str$>

In Sprix/Collage invia <str$> nel buffer della tastiera ad ogni successiva richiesta di input e senza alcuna pressione dei relativi tasti nell'ordine stabilito in <str$>; la stringa <str$> può contenere una sequenza di caratteri (es. PUTTAST "Sprix Pass") oppure di uno o più codici <cod> relativi a tasti funzione (v. SETTF) e in forma #<cod> (es. PUTTAST "#610"). Se <str$> contiene '@' seguito dal nome di una 'ETICHETTA:', sono eseguite le istruzioni a questa successive.

```
Esempio:
Collage in menu 'CA'
ON_612_CARICCOD:PUTTAST "#602@TEMPO:#501@TEMPO:#501@TEMPO:#505#505#505":
END
TEMPO: AZZCRONO 1
CICLO: IF CRONO(1)<3 THEN GOTO CICLO: END
Per evitare saturazioni del buffer di tastiera effettuare tante routine
Per inviare caratteri '#' o '@' basta ripeterli, es. PUTTAST "Spx1##Spx2@@"
```

## PUTTPLOT

In Sprix scrive i dati di Anagrafica Tipo Lotto relativi alle variabili struttura archivi _TPLOT?. Nella scrittura, il dato personalizzato si azzera se valorizzato null ""; se il Tipo Lotto è presente in archivio, i seguenti dati sono impostati con quelli in archivio:

se lo stato e 'C' creato viene mantenuto l'ordine dei dati personalizzati immessi precedentemente;

se è diverso da 'C' possiamo modifcare solo lo Stato tipo lotto la Chiave univoca solo se 'N' ed è Mantenuto l'ordine dei dati personalizzati immessi precedentemente

## RIGHT

### RIGHT$(<str$>,<numcar>)

In Sprix/Collage estrae gli ultimi caratteri <numcar> partendo da destra della stringa <str$>. Es. E$="Sprix": F$=RIGHT$(E$,3) risultato F$="rix"

```
Esempio:
G$="Sprix": H$=RIGHT$("      "+G$,7) risultato G$="  Sprix"
```

## RIGMM

### RIGMM <numriga>,<modo>,<numdecart>

In Sprix calcola i valori di riga _MM? relative ad un numero di riga valida del documento <numriga> e una testata valida, secondo un modo di calcolo <modo> (se = 1 calcola per progressivi; se = 2 calcola per totale documento; la differenza di metodo è relazionata alle righe in um2) e numero di decimali articolo <numdecart> (obbligatorio se <modo>=1).

Hanno gli stessi vincoli della PUTMM.

## SALTOPAG

### SALTOPAG$()

In Sprix/Collage ritorna una stringa di caratteri da inviare alla stampante per effettuare un salto pagina. Es. LPRINT SALTOPAG$();

## SBLOCCASEM

### SBLOCCASEM <nomesem$>

In Sprix/Collage sblocca semaforo con nome <nomesem$> bloccato da BLOCCASEM.

Attenzione, se <nomesem$>="" sono sbloccati tutti i semafori che sono stati bloccati dal proprio terminale. La semaforica per l'accesso di elaborazioni Sprix ai dati del gestionale è regolata come gli altri Client (concorrenza sul dato, regole di gestione..);la semaforica di accesso al medesimo supporto dati (es. file) è regolata da BLOCCASEM per evitare interferenze fra utenti e sovrapposizioni nell'aggiornamento dei record. Preferibili le elaborazioni in memoria per poi bloccare e scaricare su disco (evita blocchi/sblocchi continui e/o prolungati). La regola generale per l'utente è questa:

1) elaborazioni in memoria
2) blocco il semaforo
3) controllo l'errore di semaforica (anche controllo su record)
3.a) se c'è errore -> attendo
3.b) se non c'è errore -> registro dati
4) sblocco semaforo (all'evento/all'uscita del programma)
5) end/return...azzeramento memoria

```
Sprix2
BLOCCO:BLOCCASEM "*SEM1"
IF _ERRSEM THEN VIMSG 4, "Sem. bloccato da Sprix1/Collage":GOTO BLOCCO
VIMSG 4, "via libera...": SBLOCCASEM "*SEM1":END
Collage
ON_612_ALL:
BLOC:BLOCCASEM "*SEM1"
IF _ERRSEM THEN VIMSG 4, "SEM. bloccato da Sprix1/Sprix2":GOTO BLOC
VIMSG 4, "posso procedere...": SBLOCCASEM "*SEM1":END
```

# SCREEN

### SCREEN$(<nfin>,<rig>,<col>,<larg>)

Ritorna in Sprix/Collage una stringa di caratteri larga <larg>, letta a video sulla riga <rig> e colonna <col> del numero <nfin> di finestra aperta (se indicato in negativo viene inteso come ordine di apertura: 0 = base, -1 = la prima aperta, -2 = la seconda aperta,...)

# SCROLL

### SCROLL <rig>,<col>,<alt>,<larg>,<dir>

In Sprix/Collage, scrolla nella finestra attiva a partire dall'angolo superiore sinistro con coordinate numero di riga <rig> e numero colonna <col>, per un numero di righe <alt> e di caratteri <larg> verso la direzione <dir> (1= in alto; -1=in basso; 2=verso sinistra; -2=verso destra).

# SETDECFIX

### SETDECFIX <ndec>

| | |
|---|---|
| -1 | il FIX automatico a 6 decimali non viene eseguito, |
| 0-6 | rappresenta il numero dei decimali su cui deve essere eseguito il FIX automatico |

In Sprix/Collage che in base al valore <ndec> (da -1 a 6) imposta il comportamento della FIX (che altrimenti sarebbe automatico con 6 decimali su tuttii risultati derivanti da calcoli) fino alla fine del programma o fino alla riga precedente del suo riutilizzo:

Una chiamata esplicita di FIX ha priorità sul FIX eseguito automaticamente e quindi anche su quanto impostato con SETDECFIX..

```
Esempio:
SETDECFIX 2: A=70003.5399911 :
B=9337.54 : C=A+B risultato: C=79341,07 (senza FIX a 6 dec. C=79341,079999)
```

# SETFSM

### SETFSM <nfin>

Istruzione che abilita nella finestra Sprix <nfin> la gestione del mouse.

# SETMAN

### SETMAN <codvoce$>

In Sprix/Collage consente di richiamare, con tasto [F1] nell'esecuzione del campo di input del programma, la voce <codvoce$> dell'help in linea editabile con [ShF6] sul editor, che apre due voci fra cui scegliere la prima dal nome "MANUALE UTENTE" : questa apre fino a 250 voci lunghe 8 key sensitive maiuscolo/minuscolo, da potere codificare per associarvi un testo di 9000 caratteri tramite apposito editor e conteggiato alla conferma nella colonna di sinistra che viene indicizzata in ordine alfabetico. Disponibile la navigazione [frecce], cancellazione riga [ShF3], cancellazione carattere [Canc], inserimento riga/spezza riga [Invio], allineamento [F6]. Nel testo sono inseribili codici tasti funzione (elenco in SETTF) con sintassi <610>,<602>,... per disegnare i tasti funzione F10,F2,...; il codice tasto ègiusto se visualizzato in reverse. Il contenuto di codice voce <codvoce$> è richiamabile con voce o variabile (es. V$="voce": SETMAN V$: INPUT.... : INPUT ...), valida per gli input a seguire fino ad azzeramento SETMAN "" o altra SETMAN. La pressione di [ShF6] su SETMAN apre l'elenco alla voce indicata e se questa non esiste viene proposta in codifica. L'help non è con sentito in esecuzione in compilazione [ShF12]; con l'utilizzo di #APPENDISPX la definizione del<codvoce$> va nello sprix e non in libreria.

# SETNUM

### SETNUM <TipoSigla$>,<SiglaDoc$>,<Anno>,<Serie>,<Data$>,<Numeratore>

Setta il numeratore relativo all'area <Tiposigla$> ("C"=contabile,"M"=magazzino) del <SiglaDoc$> (tipo del sezionale A/V/S/C/R/X/F se l'area è "C" o la sigla del documento di magazzino se l'area è "M") riferito all'anno solare <anno> e al numero di <Serie> (1-99). In SETNUM <Data$> è quella d'aggiornamento del <Numeratore>(da 1 a 999999). Valorizza l'esito operazione in _ERRNUM$

```
Esempio: lettura di numeratore e set con nuovo valore (numero e data)
NUM = GETNUM("C","V",2009,1): IF _ERRNUM$<>"" THEN VIMSG 4,_ERRNUM$
VIMSG 1, STR$(NUM,0,0)
SETNUM "C","V",sxdaa,1,"20090508",129: IF _ERRNUM$<>"" THEN VIMSG 4,_ERRNUM$
END
```

# SETPRFIN

### SETPRFIN <numfin>,<modo>

In Sprix/Collage e nell'interfaccia grafica, setta il numero della modalità<modo> di visualizzazione

0=non proporzionale;
1=proporzionale allineato a sinistra;
2=proporzionale allineato a destra;
3 = proporzionale centrato

nel numero <numfin> finestra attiva aperta con APRIFIN (da 1-9).

# SETTATAST

### SETTATAST <tast$>,[ag$]

In Sprix/Collage setta la tastiera personalizzata dal nome <tast$> creata da apposito menu del gestionale, per essere utilizzata in modo prioritario (viene sovrascritta) rispetto a quelle predefinite (riservate e che iniziano con'_') e le personali; <tast$> non può iniziare con "_", deve essere nuovo e diverso dalla predefinita, anche se copiata/personalizzata. In Sprix il settaggio è prima di INPUT con azzeramento subito dopo; in Collage è all'evento ON_IN_? del campo di input con azzeramento all'ON_OUT_? per consentire unaltro caricamento. SETTATAST è consentita solo in Interface mode "Extend".

In Sprix, la pressione del pulsante di <tast$> copia il contenuto del campo Codice di <tast$> nel campo di INPUT e ne esce in automatico se impostato anche il campo Tasto (501 o 602)

. Esempio,
data la tastiera "tastspx":
D$="": N$="Str$": LOCATE 5,1: INPUT "A",N$,30: SETTATAST "tastspx"
N$="Str$+Tast": LOCATE 5,1: INPUT "A",N$,30: LOCATE 6,1: PRINT "DATA";:
INPUT "D",D$,10: SETTATAST "tastspx": LOCATE 6,1: PRINT "Data+Tast";:
INPUT "D",D$,10 ....

In Collage è consentita l'attivazione della modalità aggiuntiva della tastiera personalizzata: se il parametro [ag$]="a" la SETTATAST aggiunge la tastiera personalizzata a quella già presente sul campo; se il parametro [ag$]="" la SETTATAST opera in modalità sostitutiva della tastiera personalizzata al posto di quella del campo.

Esempio:
SETTATAST "tastcol","a" (l'utente può cosi' scegliere una fra le due tastiere (la "standard" e la "tastcol") sullostesso campo.

In Collage si può richiamare una tastiera nuove o predefinita personalizzata dall'utente, in cui definire pulsanti che richiamano specifiche etichette ANCHE SE NON CARICATE DA UNO SPECIFICO EVENTO, ma le regole dell'esecuzione restano immutate. Si consiglia una sola etichetta per singolo pulsante. Il richiamo all'etichetta è attraverso il campo Codice della <tast$> in cui definire "ESEGUICOLLAGE:" seguito dal nome dell'etichetta da richiamare quandoè premuto il pulsante.

Esempio.
"ESEGUICOLLAGE:ETICHETTA".
Tastiera 'tastcol' con tre pulsanti in cui uno ha 'ESEGUICOLLAGE:ETICHETTA2'
e Tastiera '_ivacod' con pulsante aggiuntivo che si attiva all'ingresso del campo iva in anagrafica articolo (E SENZA L'UTILIZZO DI SETTATAST); dati dueCollage "CollA" e "CollB" entrambi compilati nel menu 'MA'

```
            CollA                    |             CollB
-------------------------------------|-------------------------------------
ON_IN_MADSART:                       |
  SETTATAST "tastcol"                | ETICHETTA3:
ETICHETTA1:                          |  VIMSG 4, "ETIC.3 da ESEGUICOLLAGE"
  VIMSG 4, "ETIC.1 su ON_IN_MADSART"| END
END                                  |
ETICHETTA2:                          |
  VIMSG 4, "ETIC.2 da ESEGUICOLLAGE"|
END                                  |
```

# SETTF

### SETTF <str$>

In Sprix/Collage setta i tasti funzione dichiarati in <str$> nella forma di <cod>[,descrizione facoltativa fino a 20 caratteri personalizzabili] e li visualizza nella barra dei tasti funzione attivi del gestionale. I <cod> sono:

| 501 – Invio | 502 – Back space | 504 – ShTab | 505 – Esc | 506 – Ins |
|---|---|---|---|---|
| 508 – Canc | 510 – Pag.indietro | 511 – Pag. avanti | 512 – Freccia su | 513 – Freccia giù |
| 514 – Freccia sin. | 515 – Freccia des. | 516 – Shift+Su | 517 – Shift+Giu' | 518 – Shift+Sinistra |
| 519 – Shift+Destra | 520 – Shift+Backspace | 521 – Shift+Home | 522 – Shift+Fine | 528 – Shift+Canc |
| 529 – Shift+PagSu | 530 – Shift+PagGiu' | 537 – Ctrl+Sinistra | 538 – Ctrl+Destra | 539 – Ctrl+PagSu |
| 540 – Ctrl+PagGiu' | 602 – F2 | 603 – F3 | 604 – F4 | 605 – F5 |
| 606 – F6 | 607 – F7 | 608 – F8 | 610 – F10 | 611 – F11 |
| 612 – F12 (Solo Collage) | 615 – Sh+F3 | 616 – Sh+F4 | 617 – Sh+F5 | 618 – Sh+F6 |
| 619 – Sh+F7 | 620 – Sh+F8 | 623 – Sh+F11 | 624 – Sh+F12 | 625 – Ctrl+F1 |
| 626 – Ctrl+F2 | 627 – Ctrl+F3 | 628 – Ctrl+F4 | 629 – Ctrl+F5 | 630 – Ctrl+F6 |
| 631 – Ctrl+F7 | 632 – Ctrl+F8 | 633 – Ctrl+F9 | 634 – Ctrl+F10 | 635 – Ctrl+F11 |
| 636 – Ctrl+F12 | | | | |

Alcuni tasti sono riservati (F1 per SETMAN, Alt+F9 per elenco tasti attivi,...)

Esempio: SETTF "512Riga prec.,513,615"

# SETZM

### SETZM <riga>,<colonna>,<lunghezza zonamouse>,<numero zonamouse>

Istruzioni che imposta nell’input a scorrimento in una determinata zona mouse con rilascio di un determinato codice (numero zonamouse). In Collage le istruzioni valgono solo per le finestre aperte con Sprix.

# SHELL

### SHELL <comando$>

In Sprix/Collage esegue la stringa di comando <comando$> a livello di sistema operativo. L'esecuzione richiede accorgimenti in base al tipo di sistema operativo e nel rispetto dei requisiti di configurazione e sistemistici richiesti nel manuale utente. L'esecuzione di Sprix avviene con la SHELLSPRIX e la esecuzione in Collage di un menu con la ESECMENU.

Al termine dell'esecuzione del <comando$> il controllo ritorna al gestionale dal punto di esecuzione, anche senza attendere il termine dell'elaborazione del <comando$> in base a cosa si esegue: diversamente da SHELL "notepad.exe" la A$="C:\nuovacartella\": SHELL "mkdir "+A$ non richiede la chiusura di una finestra attivata dal comando. Per consentire la prosecuzione del programma dopo l'esecuzione SHELL, sistemi Microsoft Windows richiedono lo "START" nel<comando$>; senza lo "START" o con l'uso di "START /WAIT" (o "START /W" nei sistemi NT) il programma attende il termine. Es: dato un file.bat installato nella directory "C:\" (con un fittizio ping www.passepartout.net per testare la bontà della connessione ISP oppure con un semplice comando help) e data la variabile CMD$="START C:\file.bat", la SHELL CMD$ esegue senza fermarsi adifferenza di quanto farebbe con CMD$="START /W C:\file.bat".

Sistemi Windows 2000 e XP possono richiedere l'aggiunta nell'elenco di directory (in pannello di controllo - sistema - avanzate - variabili d'ambiente variabili di sistema) le directory %SystemRoot%\system32 e %SystemRoot%\system32\cmd.exe. Per eseguire programmi Windows (in VB o altro) utilizzare la SHELL con il percorso e il nome dell'eseguibile. L'unico modo di comunicare con un altro applicativo è l'utilizzo dei files.

In Client/Server l'esecuzione su Client richiede il reindirizzamento '<' del<comando$> (es. SHELL "<START /W.... ") e il default del Server si forza su Client anche con '*' (es. SHELL "*NOTEPAD"). In Linux il gestionale non reindirizza i comandi (<,>,>>) e per eseguirli occorre costruire una shell nella directory prog dell'installazione da lanciare con un altra SHELL. Ad esempioSHELL "sprixsh "+A$ (dove A$ è il comando desiderato).

Si possono settare i parametri del programma da lanciare, se sono settabili anche fuori dal gestionale. Per la definizione di directory complesse in forma abbreviata

es. SHELL "c:\PROGRA~1\MICROS~2\OFFICE10\WINWORD.EXE") la '~'si codifica con CHR$ (ALT+126 su comuni editor di testo)

# SHELLSPRIX

### SHELLSPRIX <comandospx$>

Da Sprix/Collage lancia l'esecuzione di altro Sprix tramite il <comandospx$>con sintassi dei parametri da indicare tutta su una riga nel seguente modo:

[<NomeDft>][-t<NumTerm>][-o<NumOper>][-x<Data>][-a<SiglaAz>[:<codSottAz$>]]

[-k<Ute:Pwd> o -k<Chiave>] -p[#]<Sprix> [-v<ArgCmd>].

L'unico parametro obbligatorio è l'indicazione dello Sprix -p[#]<Sprix> nella forma di numero di posizione negli elenchi del modulo Sprix (es. -p#101) oppure nella forma di nome del file -p<Sprix> senza estensione dichiarato negli elenchi del modulo (es. -pspx101) eccetto nella gestione utenti NON attivata (in cui basta l'eseguibile nella directory Sprix dell'azienda aperta).

**<NomeDft>** è il file di definizione del terminale in uso e senza estensione ".dft";

**-t<NumTerm>** è il numero di terminale da aprire, nella gestione con AUTOTERM, SHELLSPRIX apre un terminale tipo servizio, altrimenti occorre disporre di multiterminalità dove <NumTerm> assume valore numerico da 0 al massimo dichiarato nella configurazione dell'installazione;

0 collega al primo disponibile,

-2 cerca il primo terminale aggiuntivo libero

-3 cerca in modo misto, prima sugli aggiuntivi e poi quelli definiti in configurazione);

**-o<NumOper>** è il numero di operatore in luogo di quello di default,

0 si collega al primo disponibile;

**-x<Data>** è la data odierna di apertura terminale (se assente è assunta l'ultima utilizzata dal terminale o in uso, se

2 è data di sistema,

3 è l'ultimautilizzata dal terminale,

GGMMAAAA è una specifica;

**-a<SiglaAz>** è la sigla azienda da aprire, se assente è assunta quella del terminale che invoca SHELLSPRIX (:<codSottAz$> indica l'eventuale sotto azienda, v. APRIAZIE);

**-k<Ute:Pwd>** sono le credenziali di accesso agli archivi se attiva la gestione estesa degli utenti (se assente sono assunte quelle del terminale che invoca SHELLSPRIX) oppure la chiave d'accesso

**-k<Chiave>** se non è attiva la gestione estesa (in entrambe i casi lo sprix da eseguire deve essere dichiarato a menù); infine

**-v<ArgCmd>** che è la stringa da assegnare all'esecuzione dello Sprix e passata a questi prima del suo lancio tramite la variabile

# SLEEP

### SLEEP(<n>)

In Sprix/Collage sospende l'esecuzione del processo per <n> millisecondi.

Esempio
SLEEP(1500) sospende il programma per 1500 millisecondi (1,5 sec)

# SPRIGBL

### SPRIGBL <NumBL>,<NumSottoBL>,<CodPF>,<NumRifPF>,<qta1>,<qta2>,...,<qta32>

In Sprix spezza una riga di una BL in base a questi parametri obbligatori:

il numero di BL <NumBL>, numero di sottoBL <NumSottoBL>, codice PF <CodPF>, num di riferim. al PF <NumRifPF>, quantità/num. colli da spezzare <qta1> e nel caso di taglie fino alla 32^ taglia <qta32>. Il num. rif. PF è olrif() delle righe di BL: da 1-16000000 = collegata a OC, > 16000000 non collegata;la parte finale del numero (es. 1 su 16000001) indica il rif. PF. La olrig()è il num riga di BL.

La _ERRSPRIGBL e la _ERRSPRIGBL$ in Sprix riportano il codice e la descrizione dell'operazione SPRIGBL :

| | |
|---|---|
| 0 | esito positivo dell'operazione, nessun messaggio, il campo è vuoto; |
| 1 | generico, con relativa descrizione:<br>- riga OC non valida;<br>- riga già collegata a XX xx/xxxxxx;<br>- l'articolo ???? è senza DBP;<br>- codici incompatibili: '????' e '????';<br>- incompatibilità tipo articolo/quantità;<br>- documento in uso da altro terminale riga non modificabile;<br>- riga già variata da altro terminale;<br>- riga con distinta base automatica riga non modificabile;<br>- riga bloccata riga non modificabile;<br>- riga ordine cliente annullata riga non modificabile;<br>- errore generico riga non modificabile;<br>- quantità da avanzare maggiore della qta di riga sul PF; |

| | |
|---|---|
| | - quantità da avanzare pari a qta di riga sul PF;<br>- quantità da avanzare pari a zero;<br>- impossibile spezzare la riga;<br>- riga non trovata; |
| 2 | numero Bolla errato |
| 3 | memoria insufficiente |
| 4 | bolla padre non trovata |
| 5 | bolla non trovata |
| 6 | errore generico |

# SPXREMOTO

### SPXREMOTO$(<ind$>,<par1>,<par2>,<filed$>,<filee$>,<sinc>)

In Sprix (se abilitata l'assistenza remota e da assistenti verso assistiti),lancia dal Communicator Locale (Client/Server Locale) uno Sprix Remoto cioè un sevizio da eseguire lato Communicator Remoto (Client/Server Remoto o Control panel assistito) per ottenere sulla medesima installazione Locale i risultati provenienti dalle elaborazioni in Remoto (e le relative informazioni presenti lato Remoto); dal 2005 si può eseguire anche fra versioni diverse.

Lo Sprix Locale (relativo al Client/Server del Communicator Locale), lancia la SPXREMOTO$() che contatta un predefinito Server Remoto (Client/Server del Communicator Remoto) il cui "indirizzo:porta" va indicato in stringa <ind$>, per eseguire lo Sprix Remoto sul Communicator Remoto tramite la stringa di parametri di comando <par1$>:

"-aSiglaAzienda:SottoAzienda -tNumeroTerminale -xDataApertura -kUtente:Password -pNomeSprixRemoto"

(per ulteriori approfondimenti vedere APRIAZIE e SHELLSPRIX);

se Communicator Remoto gestisce Login di Mxserver aggiungere nella stringa le credenziali di accesso nella forma:

"-login=Nomeutente$ -password=Password$";

se Communicator Remoto è installazione su Server Farm Passepartout l'accesso da aggiungere è invece nella forma:

"-predesk=S -dominio=NomeDominio$ -idinst=NomeInstallazione$ -login= NomeUtente$ -password=Password$"

ed essendo queste ultime le credenziali di accesso al Predesk sostituiscono il parametro "-k". Il parametro <par2$> è la stringa (utilizzabile o meno) per l'esecuzione dello Sprix Remoto (con il"" non si passa alcunchè allo Sprix Remoto).

SPXREMOTO$() ritorna l'esito dell'elaborazione dello Sprix Remoto, riportato anche nel nome del file <filed$> che riceve eventuali errori (ad esempio nel errspx.txt si potrebbe trovare il messaggio "installazioni che non supportano questa esecuzione"); se la stringa = "" l'esecuzione è andata a buon fine. Il deposito del risultato dell'elaborazione dello Sprix Remoto è nel nome del file <filed$> che riceve i dati (esempio "datispx.txt"). Entrambi i file sono generati sul Communicator Locale attraverso apposite PUT.

Se il numero <sinc>=0 il modo di esecuzione è sincrono (SPXREMOTO$ Locale attende il trasferimento dei risultati provenienti dallo Sprix Remoto; terminata SPXREMOTO$ sono disponibili il file dati/errori); se <sinc>=1 è a sincrono (SPXREMOTO$() non attende risultati e termina, per cui occorre sapere quando trattarli).

La generazione di <filed$> e <filee$> sul Communicator Locale avvengono con PUTREM_DATI <buff$> e PUTREM_ERR <buff$>

Nello Sprix (Communicator Remoto) associato alla SPXREMOTO$() ed in grado di eseguire il servizio su sua richiesta (del Communicator Locale) inseriscono nei file dati/errori il buffer <buff$> contenente il risultato/errore della elaborazione e creano <filed$>/<filee$> nella directory dati (o dati utente) del Server Locale. ATTENZIONE: Sprix Remoto contenente le PUTREM è un'esecuzione BATCH e quindi NON deve prevedere interazioni col terminale o il video dell'utente, pena l'interruzione dell'esecuzione con notifica di errore.

Esempio:

Client (Communicator Locale) con indirizzo del suo Server "localhost:11000" vuole elaborare l'articolo "ART" sull'azienda DBL in un'altra installazione su Server Remoto (Communicator Remoto) con indirizzo "100.100.0.100:9000".

```
      -----> Sprix "spxloc.ss" su Communicator Locale <--------
P2$="ART":R$=SPXREMOTO$("100.100.0.100:9000","-aDBL -t0 -x01012009 -kUTE:PAS
   -pspxrem",P2$,"datspx.txt","errspx.txt",0)
IF R$<>"" THEN VIMSG 1, R$: END
      -----> Sprix "spxrem.ss" su Communicator Remoto <--------
 B$=_ARGCMD$: VAR$="L'argomento inviato per l'elaborazione è " + B$
 IF B$="ART_DIVERSO" THEN ERR=1
 PUTREM_DATI VAR$: IF ERR<>0 THEN PUTREM_ERR "ERRORE"
```

# SQL_ALLOCSTMT

### SQL_ALLOCSTMT <nc>,<ns>

In Sprix/Collage, nel numero connessione <nc> alloca la memoria per l'handle del <ns> numero di statement (allocazione memoria di riferimento, da 1 a 64) associato alla <nc> e che va liberato invocando SQL_FREESTMT.

Esempio

```
SQL_ALLOCSTMT 1,1 : GOSUB GEST_ERR_SQL
```

# SQL_AUTOCOMMIT

### SQL_AUTOCOMMIT <nc>,<tipo>

In Sprix/Collage, nel numero connessione <nc> imposta l'eventuale <tipo> di gestione delle transazioni (>0 è SI ovvero il default, <=0 significa NO che richiede l'invocazione di SQL_TRANSACT per chiudere la transazione). Non è utilizzabile con pxjdbc.jar poichè agisce solo in modalità AUTOCOMMIT. La operazione inversa al termine della connessione evita gli errori.

Esempio

```
SQL_AUTOCOMMIT 1,1 (senza transazione)
SQL_AUTOCOMMIT 1,0 (con transazione)
```

# SQL_BINDCOL

### SQL_BINDCOL <nc>,<ns>,<nbind>,<nomevar$>[,D]

In Sprix/Collage, nel numero connessione <nc> e di statement <ns>, assegna alla variabile <nomevar$> il valore contenuto in <nbind> numero di colonna del result set (da 1 a n = ultima colonna del recordset; tanti per quanti sono i campi del result set, da 1 a 256) all'invocazione della SQL_FETCH e che in caso di colonne di tipo DATA richiede [,D] per valorizzare la stringa in formato "AAAAMMGG". Specifica dove devono essere ritornati i risultati della lettura SELECT) da effettuare per ogni colonna del result set specificando la variabile che verrà valorizzata da SQL_FETCH.

Esempio

```
SQL_BINDCOL 1,1,1,COD      : GOSUB GEST_ERR_SQL
SQL_BINDCOL 1,1,2,DES$     : GOSUB GEST_ERR_SQL
SQL_BINDCOL 1,1,3,DAT$, D  : GOSUB GEST_ERR_SQL
```

# SQL_CONNECT

### SQL_CONNECT <nc>,<DataSource$>,<UserName$>,<Password$>

In Sprix/Collage, realizza la connessione <nc> (numero da 1 a massimo 8 contemporanee, ed è il riferimento per le informazioni sulla connessione, lo stato della transazione e le condizioni d'errore) con il database tramite il profilo di connessione <DataSource$> (nome o codice della sorgente dati per la connessione al driver ODBC per puntare al database di riferimento). Se non è specificato <UserName$> e/o <Password$> (l'utente e la sua chiave di accesso), utilizzare "".

Se in MxServer.ini il parametro LOGIN=1 occorre che gli utenti siano creati come utenti locali del Server, per l'autentificazione dell'utente all'apertura del Client, pena "Errore durante l'accesso: nome utente sconosciuto o password non valida". Se LOGIN=0 non c'è autentificazione (se si utilizza un DSN utente, questo deve essere creato dall'utente proprietario dei processi in sede d'installazione del Server; la creazione di DSN Utente con un altro nome provoca errore "Nome origine dati non trovato"; l'utilizzo di un DSN non crea problemi. Il profilo ODBC va dichiarato sul Server con i flags di sola lettura ed accesso esclusivo non impostati ed è consigliato che punti a un solo database e che sia dedicato per singola azienda. Definire il DataSource$ con il nome del DSN.

Esempio, dichiarato il DSN$="profdsn" oppure DSN$="Access"

```
SQL_CONNECT 1,DSN$,"ute","pwd" : GOSUB GEST_ERR_SQL
```

Successivamente, ma prima di eseguire ulteriori operazioni SQL, occorre dichiarare la stringa di lettura/scrittura da eseguire. L'applicazione pone la stringa SQL in una variabile alfanumerica.

Esempio

```
STR_SQL1$ = "SELECT CODICE, DESCRIZIONE, DATA FROM Tabella"
STR_SQL2$ = "SELECT * FROM Tabella"
STR_SQL3$ = "SELECT NI_CONTO,CA_NOMERIC,DT_CREAZ FROM Rubrica"
STR_SQL4$ = "INSERT INTO Tabella(Codice,Descrizione) VALUES(1,"Pippo")"
STR_SQL5$ = "INSERT INTO " +TABELLA$+" (COLONNA     ) VALUES ("+VAR$+")"
```

Query con stringa SQL > 255 caratteri richiedono di spezzare la variabile in più stringhe (es.STRSQL1$+STRSQL2$) o SQL_PREPARE per inserire più variabili concatenate direttamente da lei (es. SQL_PREPARE 1,1,STR_SQL1$,STR_SQL2$) Query con apice (') nella stringa, richiede una sua simulazione attraverso il doppio inserimento del codice ASCII CHR$(39). Ad esempio per "L'Aquila" STRSQL$="INSERT INTO Tab (Cod,Desc) VALUE(1,'L"+CHR$(39)+CHR$(39)+"Aquilà)"

Con lettere accentate in nomi tabella, controllare l'interpretazione da par te di CHR$(), se errata causa errore di esecuzione dello statement SQL.

La virgola separatore di decimali, nelle query di scrittura numerica in SQL è rappresentata dal punto (con la virgola come separatore di campi): la seguente query di inserimento in tabella del valore 2,5 nel Campo1 e del valore 1 nel Campo2 ( INSERT INTO Tabella(Campo1,Campo2) VALUES (2,5,1) ) rilascerebbe un errore di " [Microsoft][Driver ODBC Microsoft Access] Il numero dei valori nella query non corrisponde a quello dei campi di destinazione ".

Occorre pertanto trasformare il separatore da virgola a punto. Access invece riconosce il campo numerico e converte la stringa in numero, esempio, dichiarata la variabile da trattare (per la virgola) NUM$="45,6" richiede stringa STR_SQL5$="INSERT INTO PrzArt(Articolo,Prezzo) VALUES('Pippò,'"+NUM$+"')"

Per il parametro di condizione LIKE, diversamente da Access (che usa MSJET) in ambiente ODBC (standard ANSI) i caratteri jolly "*" e "?" vanno rappresentati da "%" e da "_". Query su Foglio di calcolo Excel: essendo la prima riga di Excel un'intestazione di colonna il cui contenuto è il nome della variabile inseribile all'interno della SELECT, per definire la tabella di riferimento (FROM), utilizzare la notazione CHR$(96)+NomeFoglio$+CHR$(96).

# SQL_DISCONNECT

### SQL_DISCONNECT <nc>

In Sprix/Collage chiude il numero connessione <nc> ed è obbligatoria; se è aperta una transazione associata questa viene chiusa con una ROLLBACK.

Esempio

```
SQL_DISCONNECT 1 : GOSUB GEST_ERR_SQL
 END
```

# SQL_EXECUTE

### SQL_EXECUTE <nc>,<ns>

In Sprix/Collage, nel numero connessione <nc> e di statement <ns> esegue uno statement "preparato".

Esempio

```
SQL_EXECUTE 1,1 : GOSUB GEST_ERR_SQL
```

# SQL_EXEDIRECT

### SQL_EXECDIRECT <nc>,<ns>,<str1$>[,str2$].....

In Sprix/Collage, nel numero connessione <nc> e di statement <ns> esegue l'istruzione SQL di <str1$>[,str2$]...(stringa o stringhe SQL). Non sempre è consentita (es. con accesso via pxjdbc.jar) una isolata SQL_EXECDIRECT, perchè viene richiesta SQL_PREPARE + SQL_EXECUTE (l'applicazione può scegliere di eseguire direttamente l'istruzione contenuta nella variabile oppure preparare la variabile per poi eseguirla).

Esempio

```
SQL_EXECDIRECT 1,1,STR_SQL$ :
```

# SQL_FREESTMT

### SQL_FREESTMT <nc>,<ns>

In Sprix/Collage, nel numero connessione <nc> chiude l'eventuale cursore aperto associato al numero di statement <ns>, scarta i risultati pendenti e libera tutte le risorse associate allo statement.

Esempio

```
 FINE_CONN:
 SQL_FREESTMT 1,1 : GOSUB GEST_ERR_SQL
```

# SQL_FETCH

### SQL_FETCH <nc>,<ns>

In Sprix/Collage, nel numero connessione <nc> e di statement <ns> legge una riga dal result set; per leggerle tutte in sequenza dal primo record del recordset fino all'ultimo, occorre invocarla tante volte per il num. di righe.

Viene ritornato un valore per ciascuna colonna invocata da SQL_BINDCOL per leggere le informazioni in SELECT/INTERT.

La SQL_FETCH imposta _ERRSQL_NDF che indica il fine dati nel cursore (termine del recordset). Ha valore 0 (false = righe non terminate) o 1 (true = lettura terminata).

Esempio

```
LEGGI_RECORDSET:
SQL_FETCH 1,1 : IF _ERRSQL_NDF=1 THEN GOTO FINE_CONN
PRINT STR$(COD,0,0)+" "+DESCR$
GOTO LEGGI_RECORDSET
```

# SQL_PREPARE

### SQL_PREPARE <nc>,<ns>,<str1$>[,str2$]...

In Sprix/Collage, nel numero connessione <nc> e di statement <ns>, prepara <str1$>[,str2$]... la (o le) stringa SQL per eseguire la query.

Esempio

```
SQL_PREPARE 1,1,STR_SQL$ : GOSUB GEST_ERR_SQL
```

# SQL_TRANSACT

### SQL_TRANSACT <nc>,<tipo>

In Sprix/Collage, apre automaticamente l'eventuale transazione associata al numero connessione <nc>; richiede un'operazione di accettazione modifiche di <tipo> COMMIT (accetta modifiche) se >0 o ROLLBACK (perde modifiche e riporta il database allo stato precedente l'apertura transazione) se <=0.

Esempio

```
SQL_TRANSACT 1,1 (commit)
SQL_TRANSACT 1,0 (rollback)
```

# STARTCOLLAGE

### STARTCOLLAGE <nomefile$>

In Collage carica nella memoria il nome del file Collage senza estensione <nomefile$>, che non sia già caricato e che, se assente dall'elenco dei Collage, sia nella directory sprix corrente. L'evento "ON_START:" è il primo evento processato se avviene l'effettivo caricamento in memoria.

# STOPCOLLAGE

### STOPCOLLAGE <nomefile$>

In Collage scarica dalla memoria il nome del file Collage senza estensione <nomefile$> che, se = null (""), scarica quello che ha eseguito STOPCOLLAGE e non farne eseguire più alcuna sua routine (effetto analogo a quello che si otteneva in passato con _ENDCOLLAGE=1).

# STORD

### STORD <stampante>,<serie>,<numero>,<anno1>,<anno2>

è possibile in Sprix stampare sulla stampante numero <stampante>, la storia ordine con <serie> e <numero> relativo all'anno <anno1> e <anno2> d'inizio ricerca sui movimenti di magazzino (l'evasione). La stampa equivale a quella standard del gestionale. Se gli anni sono distanti da quello in uso, si allungano i tempi di elaborazione.

STORD valorizza _ERRSTORD e _ERRSTORD$ che riportano codice e la descrizione dell'esito dell'operazione:

| | |
|---|---|
| 0 | esito positivo dell'operazione (nessun messaggio, il campo è vuoto); |
| 1 | stampa interrotta (è stato premuto il tasto ESC durante la stampa); |
| 2 | non trovati dati con la selezione specificata (non esistono documenti con i parametri indicati); |
| 3 | numero documento non valido (numero dell'ordine <=0 o > al numero massimo gestito nei numeratori dell'azienda); |
| 4 | numero serie documento non valido (num di serie = 0 o > al num. Massimo di serie gestite per l'ordine nei numeratori dell'azienda); |
| 5 | Anno non valido (anno di inizio ricerca minore o maggiore degli anni in linea gestiti dall'azienda; |

# STR

### STR$(<num>,<numdec>,<punti>)

In Sprix/Collage converte il numero <num> in stringa contenente il <num> con <numdec> decimali (da -1 a 6, se -1 è assunto automaticamente il numero di decimali di <num>) e punti separatori di migliaia (1=si, 0=no).

Esempio:

A=1234.567: B=-9876.543: STR$(A,2,1): STR$(B,1,0) = "1.234,56" e "-9876,5"

## STRCMP_CI

**STRCMP_CI(<str$1>,<str$2>)**

In Sprix/Collage confronta in modalità CASE INSENSITIVE la stringa <str$1> con la stringa <str$2>. Ritorna -1 se str$1<str$2, 0 se str$1=str$2) e 1 se str$1>str$2. Non distingue fra caratteri minuscoli e maiuscoli.

Esempio
A$="A": B$="a": STRCMP_CI(A$,B$) è 0

## STRDATA

**STRDATA$(<data$>,<forma>,<separatore$>)**

In Sprix/Collage trasforma la stringa <data$> dal formato "AAAAMMGG" (forma tipica di dato parametrico in Sprix/Collage) nel formato <forma> (1 = GGMM, 2 = GGMMAA, 3 = GGMMAAAA) e <separatore$> ("/", ".", "" ecc).

Esempio:
D1$="20090623": D2$=STRDATA$(D1$,3,"/") risultato D2$="23/06/2009"

## STRING

**STRING$(<numrip>,<str$>)**

In Sprix/Collage ripete una stringa <str$> per <numrip> volte.

Esempio
R$=STRING$(3,"a-") o R$=STRING$(3,CHR$(97)+CHR$(45)) risultato R$="a-a-a-"

## SVILDB

**SVILDB <codart$>,<codpad$>,<codcli$>,<nrord>,<qta1>,<qta2>,...,<qta32>**

In Sprix/Collage sviluppa la Distinta Base Primaria dell'articolo <codart$> e valorizza le variabili dizionario della tabella "os" (array ad accesso diretto da leggere con ciclo FOR). Altri parametri non obbligatori: il codice articolo padre <codpad$> con componente <codart$> (se <codpad$> ="" è assunto =<codart$>, ma è necessario per articoli con condizioni di tipo "A"); codice cliente <codcli$> per le eventuali condizioni e/o DBV (se = "" è assunto quello indicato in DBP); il numero di record <nrord> di riga OC (ocnri()) di cui si vuole lo sviluppo di DBVO); le quantità numeriche di suddivisione per taglia <qta1>,<qta2>..,<qta32> e per articoli non a taglie, queste vanno sostituite dal parametro <V,F> (num. di pezzi V da produrre - max 9999 – con stessa dimensione/caratteristica F, es. 10 porte da 3,5 metri: V*F-> 10,3.5) solo due valori, altri aggiunti verrebbero ignorati.

SVILDB valorizza _NSVILDB che riporta il numero di righe componenti sviluppate dalla SVILDB (se uguale a zero l'errore è contenuto nella _ERRSVILDB$). SVILDB valorizza _ERRSVILDB$ che riporta l'errore generato dalla SVILDB.

## TOTMM

In Sprix è possibile calcolare i totali del documento con TOTMM che valorizza le variabili _MM? dei totali del documento se richiamata dopo la valorizzazione delle _MM? relative ad un documento valido.

Ha gli stessi vincoli della PUTMM.

## VAL

**VAL(<str$>)**

In Sprix/Collage converte in numero la stringa <str$> a partire da sinistra terminando al primo carattere non numerico incontrato; se il primo caratteredi <str$> è "-" o "+" lo considera segno; se fra i caratteri numerici c'è un "." o "," è assunto come punto decimale (massimo 6 decimali).

Esempio
A$="1,3-": B$="12ud": C$="1A9": D$="F14": E$="+3.999": F$="-,2" : G$"1+2"
A=VAL(A$): B=VAL(B$): C=VAL(C$): D=VAL(D$): E=VAL(E$) : F=VAL(F$): G=VAL(G$)
A=1,3 : B=12 : C=1 : D=0 : E=3,999 : F=-0,2 : G=1

# VARKEY

### VARKEY <numfile>,<key$>

In Sprix/Collage nel file a indici aperto con numero <numfile> varia in base alla chiave stringa principale <key$> con i valori contenuti nelle variabili del buffer associato. In variazione se i valori delle variabili sono più lunghi dei buffer è scritta solo la parte non eccedente, diversamente (se inferiori) sono utilizzati ASCII 0 per l'eccedente ; la variazione modifica anche la chiave principale se la nuova non esiste; se la principale esiste già la variazione non è eseguita; tutte le chiavi sono obbligatorie e i loro valori non possono essere tutti ASCII 0 o ASCII 255. Se i campi contenenti chiavi sono variati i relativi file chiavi vengono riordinati. Più chiavi secondarie uguali del nuovo record vengono inserite in ordine di chiave principale. Valorizzano _NRDATI e _ERRIND.

# VIMSG

### VIMSG <tipo>,<msg$>

In Sprix/Collage visualizza il messaggio <msg$> in finestra centrale auto dimensionante (sulla lunghezza di <msg$>) di <tipo> n. 1(normale), 2 (normale)3 (aiuto), 4 (messaggio), 5 (errore). Sospende il programma fino alla pressione di Invio, il carattere "|" (ASCII 124) effettua il salto riga e il prefisso "<" o ">" allinea a sinistra/destra.

# APPENDICI SPRIX - ESEMPI

- Esempio Sprix

```
#APPENDISPX <spx901><lib2>   'libreria elaborazioni numeri e
stringhe
#APPENDISPX <lib3>           'libreria elaborazioni date e tempo
'dichiaro Variabili
 NUM_01=-45.123456 : ALFA_$="questa è la stringa 123"
'dimensiono, valorizzo e poi azzero degli Array
DIM ARR(3) 'numerico a una dimensione di 3 elementi
ARR(1)=7 : ARR(2)=28.5 : ARR(3)=1492.333
DIM ARR$(,10,20) 'alfanumerico di 3 dimensioni e la 1^ automatica
ARR$(1,1,1)="Sprix" : ARR$(2,1,1)="Collage":
ARR$(3,1,1)="MDS"'etc.
AZZARRAY ARR : AZZARRAY ARR$ ' azzero gli array
T=0 'numeratore controllo esecuzione routine
INIZIO:         1
 T=T+1   ' incremento il contatore
 IF T>0 AND T<4 THEN GOSUB CONTROL ELSE IF T=0 OR T>=4 THEN GOSUB
FINE
END
CONTROL:
IF T=1 THEN GOSUB ELABORANUMERO : GOTO INIZIO
IF T=2 THEN GOSUB ELABORASTRINGA : GOTO INIZIO
IF T=3 THEN GOSUB ELABORADATA : GOTO INIZIO
IF T>=4 THEN GOTO INIZIO
RETURN
ELABORANUMERO:
'dichiaro variabili numeriche e effettuo delle elaborazioni in
libreria
N1=252000 : N2=770.14453 : GOSUB ETIC1 : CLS
ELABORASTRINGA:
'dichiaro variabili stringa e le concateno
 A1$="Sprix;Collage;" : A2$="MDS;SDK;2009" : A$=A1$+A2$
'spezzo la stringa concatenata A$ e dimensiono Array stringa
automatico
PRINT " la lib2 spezza la stringa "+CHR$(34)+A$+CHR$(34) : PRINT
""
DIM AR$():GOSUB ETIC2A:FOR Z=1 TO I:PRINT STRING$(30," ")+AR$(Z)
NEXT
VIMSG 1, "ho spezzato la stringa Sprix;Collage;MDS;SDK;2009": CLS
GOSUB ETIC2B : CLS
ELABORADATA:
'dichiaro due date
DATA1$="20090303" : DATA2$="20090308" : GOSUB ETIC3
VIMSG 1, "ho trattato la DATA,|adesso inizio a|contare il TEMPO":
CLS
GOSUB CRONOMETRO ' conto il tempo
VIMSG 1, "ora sospendo per 2,5 secondi" : CLS
GOSUB SOSPENDO :VIMSG 1,"ho sospeso per 2,5 secondi" : CLS
ORDINA:
DIM ARRAY$(,13,4) 'il mazzo ha 13 carte per 4 semi
PRINT "": PRINT "Quanti mazzi di carte ? (1-3): ";: INPUT
"N",NUM,1
_LRECSORT=30
FOR M=1 TO NUM 'mazzi
FOR V=13 TO 1 STEP -1 'carte
FOR S=1 TO 4  'semi
IF S=1 THEN S$=" cuori  "
IF S=2 THEN S$=" quadri "
IF S=3 THEN S$=" fiori  "
IF S=4 THEN S$=" picche "
ARRAY$(M,V,S)=" Mazzo: "+STR$(M,0,0)+S$+" carta: "+STR$(V,0,0)
PRINT ARRAY$(M,V,S): PUTSORT ARRAY$(M,V,S): SLEEP 1.5
 NEXT
```

```
NEXT
NEXT
VIMSG 1, "i semi del mazzo sono|DISORDINATI||ora li|RIORDINO" :
CLS
RIPRENDI:
DIM SOSPESO$(): SS=0 : CC=0 'array sospeso per ordinare carte > 10
LEGGISORT:
IF CC=13 AND SS=4 THEN GOSUB STAMPAZ : GOTO RIPRENDI
GETSORT CARTA$ : CC=CC+1
IF CARTA$="" THEN VIMSG 1, "ho riordinato per seme": GOTO CTRL_SEM
IF VAL(MID$(CARTA$,26,2))<10 THEN PRINT CARTA$ ELSE GOSUB SOSPESO
GOTO LEGGISORT
SOSPESO:
SS=SS+1 : SOSPESO$(SS)=CARTA$ : RETURN
STAMPAZ:
FOR K=1 TO SS : PRINT SOSPESO$(K): NEXT : AZZARRAY SOSPESO$
RETURN
CTRL_SEM:
BLOCCASEM "*SEM1"
IF _ERRSEM<>0 THEN VIMSG 4,"Semaforo non bloccabile" ELSE GOTO
SCRIVI
SLEEP 1500 : GOTO CTRL_SEM
SCRIVI:
CLS: VIMSG 1,"il controllo sul blocco semaforo|mi permette di
operare"
QTA=0 : PZO=0 : TOTQTA=0 : TOTVAL=0 : S$="sprix"
PRINT " numero|quantita|prezzo "
PRINT "-------|--------|---------"
FOR I=1 TO 5 : QTA=I*12 : PZO=1+(I/3) : S$=S$+STR$(I,0,0)
PRINT "    "+STR$(I,0,0)+"    |    "+STR$(QTA,0,0)+"    | "+STR$(PZO,-
1,0)
TOTQTA=TOTQTA+QTA:TOTVAL=TOTVAL+(PZO*QTA)
NEXT
PRINT "---------------------------"
PRINT " Tot.Qta "+STR$(TOTQTA,0,0)+"    | Tot.Val. "+STR$(TOTVAL,-
1,0)
SBLOCCASEM "*SEM1"
VIMSG 1,"sblocco e libero il semaforo" : CLS
VIMSG 1,"ora cancello un carattere alla volta|dalla stringa "+S$
CICLO3:
LUNG=LEN(S$) : IF LUNG<5 THEN GOTO FINE ELSE PRINT "": PRINT "
"+S$
S$=LEFT$(S$,LUNG-1): GOTO CICLO3
FINE:
VIMSG 1,"Esempio Terminato"
END ' per non proseguire con il codice di libreria
'LIBRERIA 'SPX901' elaborazioni numeriche
ETIC1:
N=N1+N2:NEG=-
N:NN=N*1.5498:NNN=ARRO(NN,3,1):NNNN=ARRO(NN*100,3,1)/100
PRINT" N1 e N2 valgono              = "+STR$(N1,0,1)+" e "+STR$(N2,-
1,0)
 PRINT" N1 + N2 valgono           N = "+STR$(N,-1,0)
PRINT" N negativo vale         NEG =" +STR$(NEG,-1,0)
PRINT" NEG in valore assoluto      = "+STR$(ABS(NEG),0,0)
PRINT" NEG in lettere              = "+CIFLET$(NEG)
PRINT" N * 1.5498 vale          NN = "+STR$(NN,-1,0)
PRINT" NN fissato a 3 decimali     = "+STR$(FIX(NN,3),-1,0)
PRINT" NN arrotondato vale     NNN = "+STR$(NNN,-1,0)
PRINT" NN arrotondato meglio NNNN = "+STR$(NNNN,-1,0)
PRINT" NNNN in lettere             = "+CIFLETDEC$(NNNN,3)
PRINT" NN+NNNN= "+STR$(NN+NNNN,-1,0)
SETDECFIX 2: PRINT" NN+NNNN fissato a 2 dec.li = "+STR$(NN+NNNN,-
1,0)
PRINT" Carattere ASCII di 83       = "+CHR$(83)
```

```
PRINT" N Euro in $$ 
(cambio=0,5)="+STR$(CAMBIOVAL(2,5,N,"20210101",5),-1,0)
PRINT" Impaccamento binario di N  = " : IMP$=MKN$(N,5,5)
VIMSG 1, "Queste sono tutte|funzioni di trattamento VALORI 
NUMERICI"
T1$="Gli operatori ARTIMETICI|(per espressioni/concatenazioni di 
valori)|"
T2$="formano con gli operatori LOGICI e DI RELAZIONE|"
T3$="questa gerarchia decrescente di priorità d'esecuzione|"
T4$="NOT -negativo ^ * / MOD + -sottrazione < <= > >= = <> AND OR 
( )"
VIMSG 1, T1$+T2$+T3$+"|"+T4$
FINE1:
RETURN ' fine ETIC1
LIBRERIA 'lib1' elaborazioni alfanumeriche
ETIC2A: 'spezza la stringa A$ e ritorna Array per quanti sono i 
";"
I=1
CICLO1:
K=INSTR(A$,";")
IF K=0 THEN AR$(I)=A$ : GOTO SALTA
AR$(I)=LEFT$(A$,K-1) : A$=RIGHT$(A$,LEN(A$)-K) : I=I+1
IF INSTR(A$,";")=O THEN AR$(I)=A$ : GOTO SALTA ELSE GOTO CICLO1
SALTA:
RETURN' fine ETIC2A
ETIC2B: 'elaborazioni alfanumeriche
A$=A1$+A2$ : B$=AR$(1)+AR$(2)+AR$(3)+AR$(4)+AR$(5)
PRINT"A1$ e A2$ valgono  = "+A1$+" e "+A2$
PRINT"A1$ + A2$ vale A$  = "+A$
PRINT"A1$ + A2$ senza ; vale B$  = "+B$
PRINT"ASCII del primo carattere di B$ = "+STR$(ASC(B$),0,0)
PRINT"Lunghezza in caratteri di B$    = "+STR$(LEN(B$),0,0)
PRINT"Posiz. di MDS (o AR$(3)) in B$  = 
"+STR$(INSTR(B$,"MDS"),0,0)
PRINT"Pos. CaseInsensitive di Collage = 
"+STR$(INSTR_CI(B$,AR$(2)),0,0)
 PRINT"Verifica C.I. se A$ = B$        = ";:
IF STRCMP_CI(A$,B$)=0 THEN PRINT "vero" ELSE PRINT "falso"
PRINT"Converte AR$(5) in un numero    = "+STR$(VAL(AR$(5)),0,0)
PRINT"Disimpaccamento binario di N    = "+STR$(CVN(IMP$,5,5),-1,0)
PRINT"Aggiunge 3 ASCII 0 a Sprix    = ";: MK_AR$=MKA$(AR$(1),8)
PRINT MK_AR$;: PRINT "(visibile solo in debug)"
PRINT"Toglie tutti ASCII 0 a Sprix  = ";: CV_AR$=CVA$(MK_AR$,8)
PRINT CV_AR$+"(visibile solo in debug)"
PRINT"Converte B$ in maiuscolo  = "+CVMAIU$(B$)
PRINT"Converte B$ in minuscolo    = "+CVMINU$(B$)
PRINT"Ripete 5 volte Sprix   = "+STRING$(5,"Sprix")
PRINT"Estrae i primi 12 car. da B$    = "+LEFT$(B$,12)+"|"
PRINT"Estrae 7 car. da 6^ posiz.di B$ = "+"|"+MID$(B$,6,7)+"|"
PRINT"Estrae gli ultimi 10 car. da B$ = "+"|"+RIGHT$(B$,10)
PRINT"Estrae da sx e mantiene lung.B$ = ";:
PRINT LEFT$(LEFT$(B$,12)+STRING$(LEN(B$)," "),LEN(B$))+"|"
PRINT"Estrae da dx e mantiene lung.B$ = ";:
PRINT "|"+RIGHT$(STRING$(LEN(B$)," ")+RIGHT$(B$,10),LEN(B$))
VIMSG 1,"Queste sono tutte|funzioni di trattamento VALORI 
ALFANUMERICI"
FINE2:
RETURN' fine ETIC2B
LIBRERIA 'lib3' elabora date e tempo
ETIC3:
PRINT"date DATA1$ e DATA2$          = "+DATA1$+" e "+DATA2$
PRINT"data DATA1$ formattata        = "+STRDATA$(DATA1$,3,"/")
PRINT"giorni da DATA1$ a DATA2$     = 
"+STR$(DIFDATA(DATA1$,DATA2$),0,0)
PRINT"giorno settimanale di DATA2$ = "+STR$(GIORSET(DATA2$),0,0);
IF GIORSET(DATA2$)=7 THEN PRINT " (domenica)"
```

```
PRINT"DATA2$ diminuita di 3 giorni = "+PIUDATA$(DATA2$,-3)
FINE3:
RETURN 'fine ETIC3
CRONOMETRO:
FOR I=1 TO 5: AZZCRONO 1
CICLO2: IF CRONO(1)<1 THEN GOTO CICLO2
PRINT "": PRINT " Tempo in secondi = "+STR$(I,0,0): NEXT
VIMSG 1,"Conto il tempo che impiegherai|a confermare questo
messaggio"
VIMSG 4,"secondi, millisecondi trascorsi = "+STR$(CRONO(1),-1,0)
AZZCRONO 1
FINECRONOMETRO:
RETURN 'fine CRONOMETRO
SOSPENDO:
PRINT "": PRINT "      inizio  PAUSA di 2,5 secondi"
SLEEP(2500)
PRINT "": PRINT "      termine PAUSA di 2,5 secondi"
FINESOSPENDO:
RETURN ' fine SOSPENDO
```

- Esempio Sprix

Sull'utilizzo delle funzioni SETFSM, SETZM, INPUT_EXT

```
+-----------------------------------------+
|Righe modulo:                            |
|        PARAMETRI DI ESECUZIONE          |
|Nome         Descrizione          Ln  Dec|
|DAT$         Dalla Data           10   D |
|ADAT$        Alla Data            10   D |
|CTO$         Conto Contabile       9   C |
|ART$         Articolo             20   M |
|NUM          Numero a 2 decimali  10   2 |
|ALF$         Stringa              12   A |
+-----------------------------------------+
M1$="i seguenti valori sono utilizzabili con variabili nel
programma"
M2$=DAT$+" - "+ADAT$+" - "+CTO$+" - "+ART$+" - "+STR$(NUM,0,0)+" -
"+ALF$
VIMSG 1,M1$+"|"+M2$
LOCATE 10,10 : PRINT"Questo INPUT di Magazzino ha un SETMAN (premi
F1)":
LOCATE 11,35 : SETMAN "cod1" : INPUT "M",ART$,20 : SETMAN "" : CLS
_INTER=1
NF=AUTOFIN() : APRIFIN NF,1,0,2,6,15,45
DIM N$() : FOR N=1 TO 10 : N$(N)=STR$(N,0,0) : NEXT : DIM S$(10)
LOCATE 1,4: PRINT "TITOLO: finestra di PRINT e INPUT video  "
LOCATE 2,1: PRINT STRING$(45,"-")
LOCATE 3,1:COLOR 4:PRINT"Input per avanzamento..."
LOCATE 4,1 :COLOR 3:PRINT"n.0"+"1"  :LOCATE 4,6
:INPUT"AE",S$(1),10
LOCATE 4,16:COLOR 2:PRINT"n.0"+N$(2):LOCATE
4,21:INPUT"AE",S$(2),10
LOCATE 4,31:COLOR 1:PRINT"n.0"+N$(3):LOCATE
4,36:INPUT"AE",S$(3),10
LOCATE 5,1:COLOR
0:PRINT"_____________________________________________"

LOCATE 6,1:PRINT"Input tutto visibile...  "
LOCATE 7,1:PRINT"n.0"+"4":LOCATE 7,16:PRINT"n.0"+N$(5)
      LOCATE 7,31:PRINT"n.0"+N$(6)
LOCATE 8,1:PRINT"n.0"+"7":LOCATE 8,16:PRINT"n.0"+N$(8)
      LOCATE 8,31:PRINT"n.0"+N$(9)
LOCATE 9,1:PRINTF 3,1:LOCATE 9,2  :PRINTF 1,34:LOCATE 9,34 :PRINTF
9,1
     LOCATE 9,35 :PRINTF 1,10:LOCATE 9,45 :PRINTF 4,1
```

```
LOCATE 10,1:PRINTF 2,1:LOCATE 10,34:PRINTF 2,1:LOCATE 10,45:PRINTF
2,1
LOCATE 11,1:PRINTF 2,1:LOCATE 11,34:PRINTF 2,1:LOCATE 11,45:PRINTF
2,1
LOCATE 12,1:PRINTF 5,1:LOCATE 12,2:PRINTF 1,34:LOCATE 12,34:PRINTF
8,1
    LOCATE 12,35:PRINTF 1,10:LOCATE 12,45:PRINTF 6,1
LOCATE 7,6 :INPUT"AE",S$(4),10:LOCATE 7,21:INPUT"AE",S$(5),10
     LOCATE 7,36:INPUT"AE",S$(6),10
LOCATE 8,6 :INPUT"AE",S$(7),10:LOCATE 8,21:INPUT"AE",S$(8),10
     LOCATE 8,36:INPUT"AE",S$(9),10
'------
S$(10)=" proposta " : NUM=1234
LOCATE 10,2:PRINTD" n."+N$(10)+" (INPUT proposta).........:"
LOCATE 11,2:PRINTD" n.11 (SCORRE INPUT come PRINT):"
LOCATE 10,35:INPUT "AE",S$(10),10
_SCORRI=1 : LOCATE 11,38 : INPUT "N",NUM,4 : _SCORRI=0 'Input =
Print
'------
GIRO: R=0'(input C/M/ecc..) R è il flag per il giro finestra2
LOCATE 13,2:PRINT"Tipo cod. M=Mag./C=Conti/F=Finestra:
";:INPUT"AM",TIPO$,1
IF TIPO$="F" THEN GOSUB NUOVAFIN2 ELSE IF TIPO$<>"M" AND
TIPO$<>"C" THEN...
   ... GOTO GIRO
IF R=1 THEN GOTO GIRO
LOCATE 14,5 : PRINT"CODICE: ";: INPUT TIPO$,COD$,60
M1$="<Messaggio allineato a sinistra:|<<...hai inserito il codice
"
M2$="...>>|"+"|"+"ORA SETTO IL TASTO FUNZIONE|F8: Nuova Finestra
2"
VIMSG 1, M1$ + COD$ + M2$
SETTF "505Interruzione,501Prosegui,608Nuova Finestra2"
IF _TF=501 THEN GOTO LEGGIVIDEO
IF _TF=608 THEN GOSUB NUOVAFIN2 ELSE GOTO LEGGIVIDEO
LEGGIVIDEO:
LOCATE 15,5:PRINT"DATA  : ";:INPUT "D",D$,10
Y=GETPOSCURSY(1): X=GETPOSCURSX(1): DATA$=SCREEN$(NF,Y,X,10)
IF D$="" OR DATA$="" THEN DATA$="blank"
VIMSG 4, ">Messaggio allineato a destra|Data inserita = '
"+DATA$+" '"

NUOVAFIN2:
BEEP: MM1$="Emette un ' beep ' di avvertimento..."
MM2$="...e apro un altro tipo di finesta|con altri controlli"
VIMSG 2, MM1$+"|"+MM2$
NUOVAFIN2A:
_INTER=0
NF=AUTOFIN()
APRIFIN NF,4,0,6,23,7,45
SETPRFIN NF,3: SETTF
"501Invio,512Su,610Conferma,606NuovaFinestra3,505Esci"
LOCATE 1,1:PRINTD "Dalla Data     :":LOCATE 2,1:PRINTD "Alla Data
:"
LOCATE 3,1:PRINTD "Conto Contabile:":LOCATE 4,1:PRINTD "Numero 6
dec.li:"
COLOR 4
LOCATE 1,18:PRINT STRING$(10," "):LOCATE 2,18:PRINT STRING$(10,"
")
LOCATE 3,18:PRINT STRING$(25," "):LOCATE 4,18:PRINT STRING$(15,"
")
CAMPO1: LOCATE 1,18:INPUT "D",DDATA_$,10
        IF _TF=512 THEN GOTO CAMPO1
        IF _TF=501 THEN GOTO CAMPO2
        IF _TF=610 THEN GOTO CONTROLLO
```

```
        IF _TF=606 THEN GOSUB NUOVAFIN3
        IF _TF=505 THEN GOTO CHIUDIF2
CAMPO2: LOCATE 2,18:INPUT "D",ADATA_$,10
        IF _TF=512 THEN GOTO CAMPO1
        IF _TF=501 THEN GOTO CAMPO3
        IF _TF=610 THEN GOTO CONTROLLO
        IF _TF=606 THEN GOSUB NUOVAFIN3
        IF _TF=505 THEN GOTO CHIUDIF2
CAMPO3: LOCATE 3,18:INPUT "C",CONTO_$,25
        IF _TF=512 THEN GOTO CAMPO2
        IF _TF=501 THEN GOTO CAMPO4
        IF _TF=610 THEN GOTO CONTROLLO
        IF _TF=606 THEN GOSUB NUOVAFIN3
        IF _TF=505 THEN GOTO CHIUDIF2
CAMPO4: LOCATE 4,18:INPUT "N6",NDEC,15
        IF _TF=512 THEN GOTO CAMPO3
        IF _TF=501 THEN GOTO CAMPO1
        IF _TF=610 THEN GOTO CONTROLLO
        IF _TF=606 THEN GOSUB NUOVAFIN3
        IF _TF=505 THEN GOTO CHIUDIF2
GOTO CAMPO1
CONTROLLO:
 IF DDATA_$="" THEN VIMSG 5, "Campo Obbligatorio":GOTO CAMPO1
 IF ADATA_$="" THEN VIMSG 5, "Campo Obbligatorio":GOTO CAMPO2
 IF CONTO_$="" THEN VIMSG 5, "Campo Obbligatorio":GOTO CAMPO3
 IF NDEC=0 THEN VIMSG 5, "Campo Obbligatorio":GOTO CAMPO4
 IF DDATA_$>ADATA_$ AND ADATA_$<>"" THEN VIMSG 4, "date
errate":GOTO CAMPO1
 VIMSG 1, "apertura di una terza finestra"
NUOVAFIN3:
 NF=AUTOFIN()
 _INTER=0
 APRIFIN NF,1,0,14,52,4,24
SETTF"501Prosegui,505Fine"
AZZCRONO 1
CICLO:
DISTAST 1
LOCATE 1,1: PRINT "Finestra Numero = "+STR$(NF,0,0)
LOCATE 2,1: PRINT "Blocca Tasti x 5 Sec. "+STR$(S,0,0)
S=CRONO(1): IF CRONO(1)<6 THEN GOTO CICLO: AZZTAST: AZZCRONO 1
DISTAST 0
PAUSA:
LOCATE 3,1: PRINT "Premi Fine o Prosegui "
T=INKEY()
IF T<>501 AND T<>505 THEN SLEEP 100 : GOTO PAUSA ELSE GOTO
PROSEGUI
PROSEGUI:
IF T=505 THEN GOTO FINE ELSE IF T=501 THEN GOTO SCROLLA
SCROLLA: CHIUDIFIN NF:ATTIVAFIN NF-1:_INTER=1:APRIFIN
NF,1,0,3,54,9,10
RIP:X=X+1:PRINT "Sprix "+STR$(X,0,0):SLEEP 500:IF X<8 THEN GOTO
RIP
PRINT "Sprix 9";: SCROLL 1,1,19,10,1
LOCATE 9,1: PRINT "COLLAGE"
SETTATAST "tastspx": INPUT "AE",A$,7
VIMSG 1, "Imposta INPUT con quanto | stabilito dal pulsante
scelto"
CHIUDIFIN 1 : ATTIVAFIN 0 : _INTER=1 : R=0
FOR I=1 TO 15 : DISVID 1
PUTTAST "Sprix#501"
LOCATE 3+I,20+I : PRINT "Campo n."+STR$(I,0,0)+" ";: INPUT
"AM",A$,5
NEXT : DISVID 0
FINE: VIMSG 2, "FINE ESERCIZIO": END
CHIUDIF2: R=1: CHIUDIFIN 2
DDATA_$="" : ADATA_$="" : CONTO_$="" : NDEC=0
```

```
_INTER=1: ATTIVAFIN 1 : GOTO GIRO
```

- Esempio Sprix
  Sull'utilizzo della variabiel _INTEST

```
 P=0
 GIRO: VAR$="RIGA numero" : P=P+1
 IF P=4 THEN _INTEST=0 ELSE _INTEST=1
 H|A stampa carattere AUTOMATICO         pag. ~P\ (NB
saltipagina + _INTEST)
 H|N stampa carattere NORMALE
 H|E stampa carattere ESPANSO
 H|G stampa carattere GRASSETTO
 H|G
 C stampa carattere COMPRESSO
 P
 I stampa carattere CORSIVO
 S stampa carattere SOTTOLINEATO
 F invia questa riga ~P\ su File esterno
'("c:\temp\filesprix.txt"    )
 FOR R=1 TO 10
 N           ~VAR$\\\\\\\\\\ : ~R\
 NEXT
 IF P<=5 THEN GOTO GIRO ELSE CHIUDISTAMP
APRISTAMP 24,"" : N=_NUMSTAMP : LPRINT LUNGMOD$(66)
 LPRINTIMG "c:\temp\logo.bmp",0,50,20,0,0
 LPRINTBARCODE "40235028",1,1,0,15,80,30,20
 LPRINTTXT "RIGA LPRINTTXT","Courier",10,"GIS",0,15,60,"AC
C255000000"
 LPRINT INIZCO$(): LPRINT "Riga in COMPRESSO" : LPRINT FINECO$()
 LPRINT INIZES$(): LPRINT "Riga in ESPANSO" : LPRINT FINEES$()
 LPRINT INIZGR$(): LPRINT "Riga in GRASSETTO" : LPRINT FINEGR$()
 LPRINT INIZIT$(): LPRINT "Riga in CORSIVO" : LPRINT FINEIT$()
 LPRINT INIZSO$(): LPRINT "Riga in SOTTOLINEATO" : LPRINT
FINESO$()
 FOR I=1 TO 10
 LPRINT "prima LPRINT ";: LPRINT "seconda LPRINT":
IF I=3 THEN LPRINT ACAPO$():LPRINT "Alla n."+STR$(I,0,0)+" va a
capo"
IF I=5 THEN LPRINT SALTOPAG$()
IF I=5 THEN PRINTH
NEXT
CHIUDISTAMP
GETMM "MA1/271":IF _ERRMM<>0 THEN VIMSG 4, _ERRMM$
MMMODSTP$="X":_MMSXN(1)=1234:_MMSWA$(1,1)=" VAR_COLLAGE":PUTMM 4
END
                   **********************************
                   * Tabella documenti di Magazzino *
                   **********************************

TABELLA DOCUMENTI DI MAGAZZINO

       -------------------------------------------------------
       |SIGLA DOCUMENTO    |A|B|C|   D   |E| F | G|H|I |L |N|
       -------------------------------------------------------
       |IN RETTIFICHE INVEN|S|N|N|999999 |N|CF |NO|M|I+|  | |
       |FF FATTURA FORNITO.|S|N|N|99999  |S|F  |NO|M|C+|  |A|
       |FT FATTURA         |S|S|V|99999  |S|C  |NO|M|S+|  |V|
       |FS FATTURA IVA SOSP|S|S|N|99999  |S|C  |NO|M|S+|  |S|
       |FP FATTURA PROFORMA|N| | |       |S|C  |NO| |  |  | |
       |NC NOTA D~ACCREDITO|S|S|V|99999  |S|C  |NO|M|S-|  |N|
       |CO CORRISPETTIVO   |S|S|C|99999  |N|C  |SI|M|S+|  | |
       |RI RICEVUTA FISCALE|S|S|C|99999  |N|C  |SI|M|S+|  | |
       |FC RIC.FISC/FATTURA|S|S|C|99999  |N|C  |SI|M|S+|  | |
       |RS RIC.FISC SOSPESA|S|S|C|99999  |S|C  |SI|M|S+|  | |
       |SL SCARICO LAVORAZ |S|S|N|999999 |N|CF |NO|M|S+|C+| |
```

```
|CL CARICO LAVORAZ. |S|S|N|999999 |N|CF |NO|M|C+|S+| |
|DL DEPOSITO PER LAV|S|S|N|999999 |N|CF |NO|M|C-|C+| |
|RC RESO CLIENTE    |S|N|N|999999 |S|C  |NO|M|S-|  | |
|RF RESO FORNITORE  |S|S|N|999999 |S|F  |NO|M|C-|  | |
|BD BOLLA DEPOSITO  |S|S|N|999999 |N|CF |NO|M|C-|C+| |
|BF BOLLA DI CARICO |S|N|N|999999 |S|F  |NO|M|C+|  | |
|BS BOLLA DI SCARICO|S|S|N|999999 |N|CF |NO|M|S+|  | |
|BP BOLLA PROFORMA  |N| | |       |S|C  |NO| |  |  | |
|BC BOLLA ACCOMPAG  |S|S|B|999999 |S|C  |NO|M|S+|  | |
|MF ORD.NATR.FORNIT |S|N|N|65000  |S|F  |NO|A|  |  | |
|NA ORDINI NATRICE  |S|N|N|999999 |S|C  |NO|A|  |  | |
|MX ORD.NAT.PER COR.|S|N|N|999999 |S|C  |SI|A|  |  | |
|PR PREVENTIVO      |S|S|N|999999 |N|C  |NO|P|  |  | |
|PX PREV~PER.CORRISP|S|S|N|999999 |N|C  |SI|P|  |  | |
|OF ORDINE FOPNITORE|S|S|N|65000  |S|F  |NO|F|F+|  | |
|OC ORDINE CLIENTE  |S|S|N|999999 |S|C  |NO|C|O+|  | |
|OX ORDINE PER COR. |S|S|N|999999 |S|C  |SI|C|O+|  | |
-------------------------------------------------------

Colonna A
Gestione numerazione SI o NO

Colonna B
Gestione numerazione automatica SI o NO

Colonna C
Tipo serie di numerazione documento
    N non gestito
    A registro acquisti
    V registro vendite
    C registro corrispettivi
    B bolle di consegna

Colonna D
Massimo numero del documento accettato

Colonna E
Intestazione documento obbligatoria SI o NO

Colonna F
Tipo del conto nell'intestazione documento
    C = cliente
    F = fornitore

Colonna G
Prezzi con iva o senza
    SI = con iva
    NO = senza iva

Colonna H
Tipo del file su cui viene registrato il movimento.
   ( x=numero magazzino   sig = sigla azienda)
    M movimenti di magazzino      (mmaX.sig)
    P preventivi                  (prev.sig)
    C ordini clienti              (ordc.sig)
    F ordini fornitori            (ordf.sig)
    A ordini matrice              (matr.sig)

Colonna I
Tipo di movimento dei progressivi all'interno del file articoli
per tutti
i documenti che accettano un solo magazzino da cui prelevare la
merce.
(Dal magazzino: uscita della merce)
       nessuno
 I+    aumento inventarlo
```

```
   C+    aumento carico
   C-    diminuzione carico
   S+    aumento scarico
   S-    diminuzione scarico
   F+    aumento ordine fornitore
   F-    diminuzione ordine fornitore
   O+    aumento ordine cliente
   O-    diminuzione ordine cliente

  Colonna L
  Tipo di movimento de progressivi all'interno del tile articoli
per tutti
  i documenti che accettano il secondo magazzino di destinazione
della merce
  (Al magazzino: ).
  In questo caso i movimenti di magazzino registrati sul file sono
due
   -   Il primo relativo al magazzino di uscita della merce
   -   Il secondo relativo al magazzino di destinazione della
merce
  Valgono i caratteri della colonna precedente

  Colonna M
  Tipo di registrazione di primanota che genera il documento
        Nessuna
    A   fattura d'acquisto
    V   fattura di vendita
    S   fattura con iva in sospensione
    N   nota di credito
```